LOFAR - The Leiden All Sky Surveys Key Project

Documentation for Software ANAAMIKA 2.0

ANAAMIKA - a suite of fortran programs consisting of BDSM, OR_GSM, noise, qualitycontrol and analyse_islands

by

Niruj Mohan Ramanujam

Sterrewacht, Leiden University

Contents

Chapter 1

ANAAMIKA 2.0

1.1 Introduction

This manual describes a suite of programs written in fortran, named ANAAMIKA (anaamika: that which has no name; Sanskrit). This suite includes the following sets of programs :

1. Blob Detection and Source Measurement (BDSM)
2. Noise
3. Optimized Global Sky Model (OR_GSM)
4. Analyse_island
5. Qualitycontrol
6. Simulation

All of the above are bundled into a single tarball with an attached makefile. This suite of programs works only on the image domain and can be used for simple image processing tasks, source extraction, shapelet analysis, manipulating source catalogues and so on. We now briefly describe each of these tasks.

1.2 Contents

1.2.1 Blob Detection and Source Measurement (BDSM)

This is the automated source extraction program, which can be run from the command line. The input is a FITS file of an image, as well as an optional set of user-provided processing parameters. The image is then decomposed into a set of gaussians as well as shapelets, so as to minimise the emission in the residual image, consistent with noise. All relevant products are output as FITS files as well as other formats. Intermediate products can be examined in detail by the program *noise*

1.2.2 NOISE

Noise is the interactive image processing environment which can be used to view the outputs of most of the packages described here. Noise works on FITS, binary and ascii images and consists of a host of image processing tasks. It can also be used to examine BDSM output, including comparison with external catalogues and images. PGPLOT is used as the graphics environment. It also has a shapelet sub-menu which does shapelet analysis on small images.

1.2.3 Optimized Global Sky Model (OR_GSM)

This creates a Global Sky Model from input catalogues, which can be external or BDSM-created. This can be used to predict the GSM at any frequency as well as compare in detail a predicted and an observed GSM.

1.2.4 Analyse_island

Program to inspect (gaussian and) shapelet decomposition of individual islands by BDSM.

1.2.5 QualityControl

Preliminary version of quality check for input images.

1.2.6 Simulation

For a set of telescope parameters and observing parameters, this will simulate the observed sky, based on realistic source counts. This currently incorporates only single and double sources.

1.3 Installation and starting up

The whole package is distributed as a gzipped tarball. This needs pgplot, wcslib, cfitsio and a fortran compiler which works with pgplot (which means g77 3.4 or earlier for Linux and g95 for Mac. A rather simple Makefile is supplied where links to the packages above are defined. Make sure that the fortran wrappers for cfitsio and wcslib are installed (scisoft does not have them, for example, but the original installations do). Also, make sure that XSERVE, XWINDOW and NULL devices are compiled for, inside pgplot.

All the programs need working directories (*scratch*, *fitsdir*, *plotdir* and *srldir*), defined in the script makeparadefine which in turn writes out various paradefine files. All images read in are handled as double precision arrays and since the entire arrays being worked on are loaded into memory, the command 'unlimit' may help on Linux systems since it unsets the limit on stacksize. Opening a pgxwin_server before running 'noise' will also help.

1.4 Data format

All images read in are stored and worked on, as binary files in the directory *scratch* and have a *.img* extension. Ascii image files are also supported and ascii, binary and fits files can be converted from one format to another using 'noise', though of course defining FITS headers could be ambiguous in certain cases. Binary and ascii integer images are also supported to an extent (with extensions *.nmg* and *.nsc* respectively). Additionally, ascii and binary tables needed for specific tasks are also defined with specific formats. The user would rarely need to deal with these files, though.

1.5 Usage

The Makefile provided will make the executables for each of the programs mentioned above (the graphics-free BDSM needs to be compiled in seperate directory, see relevant section).

Chapter 2

Blob Detection and Source Measurement (BDSM)

2.1 Introduction

f-BDSM is written in fortran 77 and is a prototype for the final Python based PyBDSM which is currently being developed. This has been tested on simulations and real data (including CS1) and the report of these tests can be found elsewhere. f-BDSM (latest version 2.0) is currently maintained by the author at Leiden. Though BDSM can be run from the command line, all the tasks inside this package are also available individually in the package *noise* . The automatised graphics-free executable is called bdsm.

2.2 Contents and installation

BDSM is a part of ANAAMIKA and is very easy to use; even the author can use it. Each individual module of BDSM can be run seperately inside *noise* but we will now describe the working of the automated graphics-free command-line version.

The mother package ANAAMIKA has its source files in the main directory ANAAMIKA. In a subdirectory called 'bdsm-dir' is a script called 'create', which when run, copies all the fortran files and strips them of all lines of code which use the graphics package pgplot. This is done so that users who do not have pgplot installed and are not interested in running *noise* can run BDSM as a stand-alone software. Note that pgplot no longer compiles and runs properly with fortran compilers later than 3.4. Following this, the command 'make bdsm' will create the executable 'bdsm', which needs only a fortran compiler, cfitsio and wcslib.

2.3 What the automated BDSM does for you

BDSM performs the following in order -

- **Reads initial parameters** from 'paradefine'.
- **Reads in FITS file** *(callreadfits.f)*. This uses *cfitsio* to read in the FITS file, assumed to have only 2 dimensional data (3^{rd}D can be defined as long as its one pixel wide). Writes out the image to the scratch dir as a binary file ($scratch/<imagename.img) and the header (including the beam sizes which annoyingly seem to be commented in AIPS files rather than put in as keywords) information to an ascii file ($scratch/*imagename.solnname.header*).

- **Calculates basic statistics of the image** *(preprocess.f)* and stores them in the file $scratch/*imagename.solnname.bstat*. Calculates sensible values of processing parameters and stores them in *scratch/imagename.solnname.bparms*. First we calculate mean and rms, with and without (3σ) clipping, min and max pixel and values, solid angle. Hereafter, rms indicates the 3σ clipped measure. Next, the number of beams per source is calculated (see A1), using which a sensible estimate of *boxsize* and *stepsize* are calculated (see A1). Finally, the thresholds are set. They can either be hard-thresholded (by the user or set as 4σ for pixel threshold and 3σ for island boundaries internally) or can be calculated using the False Detection Rate (FDR) method using an user defined value for α. If the user does not specify whether hard thresholding or FDR should be applied, one or the other is chosen internally based on the ratio of expected false pixels and true pixels (the choice is written out in *imagename.solnname.bparms*).

- **Calculates rms image** *(callrmsd.f)*. 3σ clipped rms and mean are calculated inside boxes of size *boxsize* in steps of *stepsize*. Intermediate values are calculated using bilinear interpolation (it was seen that bicubic spline did not yield appreciably better results but is also available). Depending on the resulting statistics (see A1), we either adopt the rms image or a constant rms in the following analysis.

- **Identifies islands of contiguous emission** *(formislands.f)*. First all pixels greater than the pixel threshold are identified (and sorted by descending flux order). Next, starting from each of these pixels, all contiguous pixels (defined by 8-connectivity, i.e., the surrounding eight pixels) higher than the island boundary threshold are identified as belonging to one island, accounting properly for overlaps of islands.

- **Internal bookkeeping** *(iland_mat2list.f)* where the file storing the identified islands is written out in a more convenient format for use by the program.

- **Fit multiple gaussians and/or shapelets to each island** *(sourcemeasure.f)*. For each island, the subimages of emission and rms are cut out. The number of multiple gaussians to be fit can be determined by three different methods. The default method is described in A1. With initial guesses corresponding to these peaks, gaussians are simultaneously fit to the island using the Levenberg-Marqhardt algorithm. Sensible critera for bad solutions are defined (see *soln_quality*). If multiple gaussians are fit and one of them is a bad solution then the number of gaussians is decreased by one and fit again, till all solutions in the island are good (or zero in number, in which case its flagged). After the final fit to the island, the deconvolved size is computed assuming the theoretical beam and the statistics in the source area and in the island are computed and stored. Errors on each of the fitted parameters are computed using the formulae in Condon (1997). Finally all good solutions are written into the gaussian catalog as an ascii and binary file. If shapelets are required, the program calculates optimal *nmax*, β and the centre, and stores these and the shapelet coefficients in a file. Currently these coefficients are calculates by integration, but we will fit then instead in the next version.

- **Compute residual image** *(make_src_residim.f)*. The source list is read again (and hence can be modified/purged before this if required) and the residual image is computed. Both the gaussian and shapelet decomposed residual images are written.

- **Write out the residual image** *(callwritefits.f)* as a FITS file.

- **Group gaussians into sources** *(callgaul2srl.f)* and output the source catalog. This is by nature a bit ad hoc, and is described in Sec. 2.7, para 5.

- **Write out the gaussian and source catalogs as FITS binary tables** *(callwritegaulfits.f and callwritesrlfits.f).*

2.4 Usage

The input parameters for *bdsm* are written in the file 'paradefine'. This file is created by running the script 'makeparadefine' and all paths should be defined in this script for your hostmachine. The resulting file 'paradefine' includes the directory paths for output files as well as source extraction parameters which can be specified by the user. Letting the default values remain leads to the program calculating sensible values by itself (which can later be seen in the file '<imagename>.bparms'.

The variables defined in the file 'paradefine' are the following, some of which are explained in detail in Sections 2.3 and 2.7.

- fitsdir - directory for input and output residual FITS files
- scratch - scratch directory for .img and other files
- srldir - directory for source and gaussian lists
- plotdir - directory for storing ps files of plots
- fitsname - name of fits file (<imagename> is the .fits/FITS stripped fitsname)
- solnname - identifier for this run of bdsm. Most files are named <imagename>.<solnname>.*
- runcode - (a)uto/(m)anual and (q)uiet/(v)erbose, e.g. 'aq'
- gausshap - '11' fit gaussian, shapelets (1=fit; 0=dont fit); default=11=both
- bmpersrc_th - beams per source, 0 → calculated inside bdsm
- boxsize_th - boxsize for rms image, 0 → calculated inside bdsm
- stepsize_th - stepsize for rms image, 0 → calculated inside bdsm
- thresh - 'hard' or 'fdr' pixel threshold, 'default' → calculated inside bdsm
- fdr_alpha - alpha value for FDR; 0 → set to 0.05 inside bdsm
- thresh_pix - pixel threshold (in terms of local rms), 0 → set to 4 inside bdsm
- thresh_isl - island threshold (in terms of local rms), 0 → set to 3 inside bdsm
- minpix_isl - minimum number of pixels per island, 0 → set to 4 inside bdsm
- rms_map - take 'const'ant rms or rms 'map', 'default' → calculate inside bdsm
- ch0plane - plane of 3d image to extract from, 0 & invalid → all channels collapsed
- maxsize_beam = 0 ; maximum size of fitted gaussian in beam area, 0 → set to 10 inside prog
- takemeanclip = 'default' ; take calculated mean_clip or 0, not true/false → calculate in prog
- BMAJ = 0.0 ; BMAJ (fwhm deg). If not in fits header, take this, and if bmaj=0, ask user
- BMIN = 0.0 ; BMIN (fwhm deg). If not in fits header, take this, and if bmin=0, ask user
- BPA = 0.0 ; BPA (deg). If not in fits header, take this

The program is now run by invoking the executable 'bdsm'. All images are handled as double precision arrays and at a given time, the program will handle no more than two copies of the image, which should tell you the size of memory required (*unlimit* helps). The names and locations of the outputs are stored in a file whose name is printed on the screen.

2.5 Output

The primary outputs are the residual images (as FITS files) and the gaussian and source lists and the shapelet parameters and coefficients. These lists are available as an ascii file, a FITS binary table, and an AIPS-readable 'star' file and a ds9-readable 'region' file for the gaussians. The ascii lists have a few headers, and the fields following these are described below.

All the intermediate products are either ascii or binary files. The latter can be visualised using *noise* including overlaying island boundaries on the image, inspecting initial guesses for gaussian fits, the background rms and mean images, and so on. Note that BDSM and *noise* use the same modules and hence the same data formats and hence the latter is indispensable for troubleshooting the output of BDSM.

2.5.1 List of output files

The list of output files are :

- Freq collapsed image (FITS) = $fitsdir/imagename.solnname$.ch0.FITS
  This is the frequency collapsed 2d FITS image, which is of course the same as the original image if the latter has only one channel. If not, the cube is collapsed using the value of *ch0plane* in the file 'paradefine', and this collapsed image is what BDSM is run on.
- Gaussian subtracted residual image (FITS) = $fitsdir/imagename.solnname$.resid.gaus.FITS
  This is the original FITS image after subtracting the fitted gaussians.
- Gaussian catalogue (ascii) = $srldir/imagename.solnname$.gaul
  This is the catalogue of fitted gaussians in ascii form. The format is given below.
- Source catalogue (ascii) = $srldir/imagename.solnname$.srl
  This is the catalogue of the sources identified by combining gaussians, in ascii form. The format is given below.
- Gaussian catalogue (FITS) = $srldir/imagename.solnname$.gaul.FITS
  This is the gaussian list as a FITS binary table, readable by external software, e.g. fv.
- Source catalogue (FITS) = $srldir/imagename.solnname$.srl.FITS
  This is the source list as a FITS binary table, readable by external software, e.g. fv.
- Gaussian STAR file (AIPS) = $srldir/imagename.solnname$.gaul.star
  This is the gaussian catalogue as an AIPS-readable STAR file.
- Gaussian reg file (ascii) = $srldir/imagename.solnname$.gaul.reg
  This is the gaussian catalogue as an ds9-readable .reg file.
- Source STAR file (AIPS) = $srldir/imagename.solnname$.srl.star
  This is the source catalogue as an AIPS-readable STAR file.
- Shapelet reconstructed source image (FITS) = $fitsdir/imagename.solnname$.srcim.shap.FITS
  This is the FITS image reconstructed using the fitted shapelets. The pixels outside islands are set to NaN and the pixels inside the island cutout (formed by blc, trc) but not inside the island are set to zero.
- Shapelet subtracted residual image (FITS) = $fitsdir/imagename.solnname$.resid.shap.FITS
  This is the FITS image formed by subtracting the reconstructed shapelets from the original image.
- Shapelet parameter catalogue (FITS) = $srldir/imagename.solnname$.shapelet.c.FITS
  This is the FITS binary table of the shapelet parameters for each island. The format is described below.
- Shapelet parameter catalogue (ascii) = $srldir/imagename.solnname$.shapelet.c
  This is the ascii catalogue of the shapelet parameters for each island. The format is described below.

- Shapelet coefficients (ascii) = *srldir/imagename.solnname*.shapcoef.c
  This is the ascii catalogue of the shapelet coefficients for each island, corresponding to the parameters listed in the previous file. The format is described below.

- Image statistics (ascii) = *scratch/imagename.solnname.bstat*

- Image processing parameters (ascii) = *scratch/imagename.solnname.bparms*

2.5.2 Format of gaussian catalogue

The columns of the gaussian catalogues are listed below. The ascii file has a list of headers at the beginning which the binary file and the FITS binary table do not. The format of each row is also listed as one of the headers.

- Gaussian number - serial number of the gaussian
- Island number - serial number of the island the gaussian belongs to. Islands can be fit with multiple gaussians (island information is stored in $scratch/<imagename>.<solnname>.islandlist.asc
- Flag - if flag is zero, the gaussian is valid. The (nonzero) value of the flag indicates the nature of the error, defined as flag=$\Sigma\ 2^n$, for the following conditions.
  - n=0 : peak too low; peak $< thresh\_pix \times \sigma_{\rm local}$), $thresh\_pix$ is defined in paradefine.
  - n=1 : peak too high; peak > 2×maximum value in island),
  - n=2 : x position outside island),
  - n=3 : y position outside island),
  - n=4 : very large major axis; fwhm of fitted major axis in pixels $> \sqrt{maxsize\_beam} \times$ beam_fwhm$_{\rm pix}$, where $maxsize\_beam$ is defined in the file paradefine),
  - n=5 : very small major axis; fwhm of fitted mmajor axis in pixels $< 0.5 \times$ beam_fwm$_{\rm pix}$,
  - n=6 : very large minor axis; similiar to the condition for n=4,
  - n=7 : very small minor axis; similiar to the condition for n=5,
  - n=8 : cannot get sensible sky coordinates for peak pixel using wcslib).
- Total flux, in Jy
- Error in total flux, in Jy
- Peak flux deinsity, in Jy/beam
- Error in peak flux density, in Jy/beam
- R.A. of centre of fitted gaussian, in degrees
- Error in R.A., in degrees
- Dec of centre of fitted gaussian, in degrees
- Error in Dec, in degrees
- X-position of centre of fitted gaussian, in pixels
- Error in x-position, in pixels
- Y-position of centre of fitted gaussian, in pixels
- Error in y-position, in pixels
- FWHM of major axis of fitted gaussian, in arcsec
- Error in fwhm of major axis, in arcsec
- FWHM of minor axis of fitted gaussian, in arcsec
- Error in fwhm of minor axis, in arcsec
- Position angle of fitted gaussian, east from north, in degrees
- Error in P.A., in degrees

- FWHM of major axis of gaussian resulting from deconvolving gaussian synthesized beam from fitted gaussian, in arcsec
- Error in fwhm of deconvolved major axis, in arcsec
- FWHM of minor axis of gaussian resulting from deconvolving gaussian synthesized beam from fitted gaussian, in arcsec
- Error in fwhm of deconvolved minor axis, in arcsec
- Position angle, east from north, of gaussian resulting from deconvolving gaussian synthesized beam from fitted gaussian, in degrees
- Error in deconvolved P.A., in degrees
- RMS in the source area in the residual image
- Mean in the source area in the residual image
- RMS in the island area of the gaussian in the residual image
- Mean in the source area of the gaussian in the residual image
- Chisq (meaningless now) followed by its q value (also meaningless now)
- Source id - the serial number of the source which is constituted by this gaussian, in the source list *srldir*/<imagename>.<solnname>.srl
- blc1 - x-coordinate of bottom left pixel of cutout enclosing the island
- blc2 - y-coordinate of bottom left pixel of cutout enclosing the island
- trc1 - x-coordinate of top right pixel of cutout enclosing the island
- trc2 - y-coordinate of top right pixel of cutout enclosing the island

2.5.3 Format of source catalogue

The columns of the source catalogues are listed below. The ascii file has a list of headers at the beginning which the binary file and the FITS binary table do not. The format of each row is also listed as one of the headers. The algorithm for grouping gaussians into sources is described later in this chapter.

- Source number - serial number of the source
- Island number - serial number of the island the source belongs to. Islands can be fit with multiple sources (island information is stored in $scratch/<imagename>.<solnname>.islandlist.asc
- Flag - if flag is zero, the source is valid. The (nonzero) value of the flag is the sum of the flag values of the constituent gaussians the source is made of.
- Code - 'S' stands for single isolated source fit with one gaussian, 'C' for a source fit with one gaussian but is inside an island which was fit with multiple gaussians (hence this island will have more than one source) and 'M' is an extended source fit with multiple gaussians. single sourceTotal flux, in Jy
- Total flux, in Jy
- Error in total flux, in Jy
- Flux density of the peak, defined as the centroid of emission, in Jy/beam
- Error in centroid-peak flux density, in Jy/beam
- Flux density of the peak, defined as the position of maximum intensity, calculated by fitting a single gaussian to one synthesized beam area around the maximum intensity pixel, in Jy/beam
- Error in maximum-peak flux density, in Jy/beam
- R.A. of the centroid-peak position, viz. first moment, in degrees
- Error in centroid-peak R.A., in degrees
- Dec of the centroid-peak position, viz. first moment, in degrees
- Error in centroid-peak Dec, in degrees
- R.A. of the maximum-peak position, in degrees
- Error in maximum-peak R.A., in degrees
- Dec of the maximum-peak position, in degrees
- Error in maximum-peak Dec, in degrees

- FWHM of major axis of source, calculated from moments, in arcsec
- Error in fwhm of major axis, in arcsec
- FWHM of minor axis of source, calculated from moments, in arcsec
- Error in fwhm of minor axis, in arcsec
- Position angle of source, calculated from moments, east from north, in degrees
- Error in P.A., in degrees
- FWHM of deconvolved major axis of source, calculated by deconvolving synthesized gaussian beam from gaussian model of source formed using moments, in arcsec
- Error in deconvolved fwhm of major axis, in arcsec
- FWHM of deconvolved minor axis of source, calculated by deconvolving synthesized gaussian beam from gaussian model of source formed using moments, in arcsec
- Error in deconvolved fwhm of minor axis, in arcsec
- Deconvolved position angle of source, calculated by deconvolving synthesized gaussian beam from gaussian model of source formed using moments, east from north, in degrees
- Error in deconvolved P.A., in degrees
- RMS in the island area of the gaussian in the residual image
- Number of gaussians - the number of gaussians the source is constituted from
- blc1 - x-coordinate of bottom left pixel of cutout enclosing the island
- blc2 - y-coordinate of bottom left pixel of cutout enclosing the island
- trc1 - x-coordinate of top right pixel of cutout enclosing the island
- trc2 - y-coordinate of top right pixel of cutout enclosing the island

2.5.4 Format of shapelet files

Shapelet analysis of each island results in two shapelet parameter files, as described above, namely *srldir/imagename.solnname.shapelets.c* and *srldir/imagename.solnname.shapcoef.c*. The *.c* suffix indicates that the shapelets are cartesian, as opposed to polar. The algorithms for calculating these parameters are described later. The columns of the .shapelets.c file are :

- Island number - serial number of the island being decomposed
- nmax - maximum order of shapelet functions the island is decomposed into. Since the image is 2d, the (x,y) indices of the cartesian shapelets go upto (n1,n2) such that n1+n2=n_{max}
- beta - the scale β of the shapelet decomposition, the same for both x and y directions
- xcentre - the x coordinate of the centre for the shapelet decomposition
- ycentre - the y coordinate of the centre for the shapelet decomposition
- std_res - rms of the residual in the island
- av_res - mean of the residual in the island
- blc1 - x-coordinate of bottom left pixel of cutout enclosing the island
- blc2 - y-coordinate of bottom left pixel of cutout enclosing the island
- trc1 - x-coordinate of top right pixel of cutout enclosing the island
- trc2 - y-coordinate of top right pixel of cutout enclosing the island

The .shapcoef.c file is an ascii file having as many lines as the number of islands. Each line has the island number followed by the list of all shapelet coefficients for that island corresponding to the parameters in the .shapelets.c file. This is also written as a FITS binary table. Since the coefficient matrix is lower triangular, only the $n_{max} \times (n_{max}-1)/2$ numbers are stored, in fortran order.

2.6 List of non-standard data products of BDSM

Here we list all the data products of BDSM, which are not part of the standard list output to the user. These are in *noise* format files, used internally, which the user normally need not consult, unless troubleshooting. All files ending with *.img* are binary images which can be visualised and manipulated inside *noise* .

The files in the *scratch* directory are the following, and it is assumed that all of these filenames are prefixed by *imagename.solnname.*

- ch0.img - binary file of the channel collapsed 2d-image.
- header - ascii file of all the headers in the FITS file.
- islandlist - binary file listing all the pixels in each island, which can be overplotted on an image using *noise* .
- islandlist.asc - ascii file of the same. The first line has n, m, nisl and maxmem where (n,m) is the image size, nisl is the number of islands and maxmem is the maximum number of pixels in any island. Following this, for every island, there are two rows, the first listing the number of the island, the number of pixels in it, the x and y coordinates of the peak pixel and its value. The second line has all the (x,y) coordinates listed.
- mean.img - the image of the background mean, using boxsize and stepsize.
- rmsd.img - the image of the background std, using boxsize and stepsize.
- norm.img - the image, normalised by the background mean and std.
- rank.img - an image file where pixels not in islands have a value zero and island pixels have a value which is the island number.
- rank.nmg - same as rank.img except that the values are integers.
- resid.gaus.img - residual image after subtracting gaussians. Also written as a FITS file.
- resid.shap.img - residual image after subtracting shapelets. Also written as a FITS file.
- snum - internal file indexing gaussians taking into account multiple gaussians per island.
- srcim.gaus.img - binary image of all the reconstructed gaussians.
- srcim.shap.img - binary image of all the reconstructed shapelets.

The following are in *srldir* and have the same prefix as the previous list.

- gaul.bin - binary version of the .gaul file. Is used for overplotting by *noise* .
- ini.gaul - catalogue in the same format as .gaul but with the initial guess gaussians used for fitting. Can be overplotted using *noise* .
- ini.gaul.bin - binary version of above.
- srl.bin - binary version of the .srl file.

2.7 Details of certain BDSM algorithms

1. Number of beams per source, *bmpersrc* is calculated assuming the number of sources in the image, N_s, as
   N_s = (No. pixels $> 5\sigma$)/(<*pix*/*src*>),
   where the average number of pixels per source, <*pix*/*src*>, is given by
   $2\pi\sigma_{\rm major}\sigma_{\rm minor} \times (ln(S_{\rm min}/5\sigma) - 1/(\alpha - 1))$,
   where α is the slope of the differential number counts (value from Katgert et al). Assuming a minimum of one pixel to define a source and ignoring the effect of noise for sources close to the threshold, we can ignore the logarithmic term and hence

$bmpersrc$ = n×m/(No. pixels $> 5\sigma$)/((α-1). The value of $bmpersrc$ is used to decide whether the image is expected to be confused (and if so, the mean image is taken to be zero) and also to estimate boxsize for calculating the rms image (see below).

2. The boxsize and stepsize for calculating the rms image are estimated as follows.
   Typical intersource seperation, s_1, is $2\sqrt{bmpersrc} \times B_{\rm major}$.
   Size of brightest source, $s_{\rm max}$, is $2B_{\rm major} \times \sqrt{2ln(Max_{flux}/threshold)}$.
   Boxsize is estimated as the geometric mean of the larger of the quantities (s_1 and $s_{\rm max}$) and a fourth of the smallest dimension of the image. Stepsize is then calculated as the minimum of a third of boxsize and a tenth of the smallest image dimension. These prescriptions yield reasonable numbers for the images tested, but a more rigorous method would have to look at the power in different fourier modes of the image.

3. Either the calculated rms image or a constant rms is used for subsequent analysis based on whether the dispersion in the rms image is consistent with, or is higher that, the expected statistics. Hence if the dispersion of the rms image is higher than 1.1 times (clipped) rms of the image times the inverse of $\sqrt{2}\times{\rm Boxsize_{pixels}}$ then the rms image is taken else the constant value of the clipped rms is used.

4. The current procedure for calculating the number of gaussian to be simultaneously fit to an island is as follows. First, the number of gaussians is identified with the number of distinct peaks (higher than the pixel threshold) of emission inside the island (negative gradient in all 8 directions). These peaks are CLEANed from the subimage assuming the theoretical beam. If the (unclipped) rms of the residual subimage is greater than the (clipped) rms in the region, the maximum pixel in the residue is greater than the threshold for this former rms, and is located atleast 0.5 beams (and $\sqrt{5}$ pixels) away from all previous peaks, then this residual peak is identified as a new one.

5. Inside each island, groups of gaussians are deemed to be a part of the same source if 1. no pixel on the line joining the centres of any pair of gaussians has a (gaussian reconstructed) value less than the island threshold, *and* 2. the centres are seperated by a distance less than half the sum of their fwhms along the PA of the line joining them. For each source, parameters are calculated and output in the sourcelist (.srl) file.

6. For each island, a shapelet decomposition is done after estimating the best values of the centre, the scale β, and $n_{\rm max}$ in the following way. First, an initial guess of β is taken as $2\sqrt{m_2(x)m_2(y)}$ where m_2 is the second moment over the island, based on shapeelt analysis of simulated images of resolved sources. Similarly a guess for $n_{\rm max}$ is taken as the minimum of 14, and maximum of 10 and $2n+2$ where n=$\sqrt{n^2+m^2}/n_p^m - 1$ where (n,m) is size of the island and n_p^m is the synthesized beam minor axis fwhm in pixels. This guess for $n_{\rm max}$ is based partly on simulations and partly on the requirememts of computing time, number of constraints etc for shapeelt decomposition.

   These initial values are then used to calculate the optimal central position around which to decompose the island. First, for every pixel in the island, the coefficients c_{12} and c_{21} are computed assuming that pixel as the centre of expansion. Next, the zero crossings for every vertical line of the c_{12} image and horizontal line of the c_{21} image are computed. The intersection point of these two zero-crossing vectors is then taken as the proper centre of the expansion for the image. If this procedure does not work, then the first moment is taken as the centre.

   This updated centre position is used to compute the optimal β, which is taken as the value of

β which minimises the residual rms in the island area. Using this β, the centre is computed once more and the final shapelet deocmposition is then made.

Chapter 3

NOISE

3.1 Introduction

noise is the name of the interactive package for image analysis, written in fortran and uses pgplot for graphics.

This package works on images which can be read in from FITS files. Images can also be created inside the package using simple tasks. Various mathematical operations can then be performed on these images (basic statistics, image addition, median filtering etc, see below for the full list). They can also be displayed, with histograms, slice plots etc. Related to this package is Blob Detection and Source Measurement (BDSM), described in the previous chapter BDSM outputs a list of gaussians (and sources, and a residual image) given an input image. 'Noise' can then be used to overplot these gaussians on the image and analyse the products of BDSM. It can also be used to associate multiple source lists and analyse the associations (plot derived quantities against each other and overplot associated sources over the image etc).

3.2 Contents, installation and starting up

Though the requirements for this package overlap with those of BDSM, we repeat them here for convenience. This software needs pgplot, wcslib, cfitsio and a fortran compiler which works with pgplot (which means g77 3.4 or earlier for Linux and g95 for Mac. A rather simple Makefile is supplied and 'make noise' does the job. Make sure that the fortran wrappers for cfitsio and wcslib are installed (scisoft does not have them, for example, but the original installations do). Also, make sure that XSERVE, XWINDOW and NULL devices are compiled for, inside pgplot.

'Noise' defines four working directories – scratch, srldir, plotdir and fitsdir, defined in the file 'paradefine' which in turn can be written out using the script 'makeparadefine'. These directory paths should be defined in 'makeparadefine'. Note that the remaining parameters in paradefine are necessary only for the command-line bdsm and can be ignored here. The program is now run by invoking the executable 'noise'. All images are handled as double precision arrays and since the entire arrays being worked on are loaded into memory, the command unlimit may help. Opening a pgxwin_server before running 'noise' will also help.

When 'noise' is executed, you get the 'noise'-prompt which is ' Option —>' and a window is opened with a menu of tasks, each described by a two-letter word and a brief description of what it does. Each task is invoked by typing in the two-letter word at the prompt. You will then be asked for inputs pertaining to that task. Note that there are also sub-menus accessible from inside, namely *sm* which is the interactiver version of f-BDSM and *ta* which is a set of tests. Shapelets

and simulations (which is currently another package defined in the same Makefile) will soon be integrated as sub-menus shortly.

3.3 Data format

All images inside 'noise' are stored and worked on, as binary files in *scratch* and have a *.img* extension. These *.img* images can either be created afresh (option *cn*) or read in from a FITS file (option rf). Ascii image files are also supported and ascii, binary and fits files can be converted from one format to another (option ff), though of course defining FITS headers could be ambiguous in certain cases. Binary and ascii integer images are also supported to an extent (with extensions *.nmg* and *.nsc* respectively). Additionally, ascii and binary tables needed for specific tasks are also defined with specific formats. The user would rarely need to deal with these files, though. Currently, except for certain specific tasks like reading in fits files, most tasks will work only on 2d images.

3.4 Usage

Here, we briefly describe each task in 'noise' and its sub-menus. Note that except for the input/output tasks, the extension is never specified (it is assumed to be *.img* or as appropriate).

3.4.1 Book-keeping

- *li* : **Lists all binary images** ($*.img$) along with the path (*scratch*) with the latest file listed at the end (ls -ltr).
- *la* : **Lists all files** in *scratch* in similar order.
- *df* : **Deletes files.** Prompts for an input which is a, i, or f which deletes all files in *scratch*, all *.img* files in scratch or a particular file whose name you input without the path but including the extension.
- *rh* : **Obsolete.**
- *ph* : **Obsolete.**
- *ch* : **Obsolete.**
- *ex* : **Experimental tasks.** When new tasks are coded and tested, this template task can be used.
- ff : **File format converter.** Can convert a binary or ascii image file to a ascii, binary or fits file. Name of the file needs to be input without the path but with the extension. If it recognises the format (which basically depends on the code recognising the correct format in your OS using the command 'file'), then it prompts for a code - b2a means binary to ascii for example. It then gives you the output file name.
- *qn* : **Quits** 'noise', killing all the menu and sub-menu windows.

3.5 Display image

This is the main task for displaying images, coded as 'di', and has many options, described below. Given the image name, it opens up a sub-menu in another window each of whose options can be clicked on with the mouse.

1. **Zoom in :** A pair of vertices defining a diagonal of a rectangle can be clicked on. Greyscale is preserved.
2. **Zoom out :** Zooms out to the full image, clears all the overlaid graphics (except the image itself). Greyscale is preserved.
3. **Greyscale :** Switches to greyscale representation of the image and is the default.
4. **Colour :** Switches to colour for the image display. It opens up a sub-menu with some well known colour map options to choose from and then transfers control back to the main display menu.
5. **Imagescale - histogram :** The greyscale (also for the colour) can be set by defining the range on the histogram, which is shown at the bottom.
6. **Imagescale - area :** This enables defining a box on the image by clicking on the two vertices of a diagonal and the minimum and maximum inside the image is set as the limits of the new grey/colour scale.
7. **Imagescale - Pixrange :** The grey/colour scale is set by typing in the intensity range at the 'noise' prompt. Note that even if the intensity is displayed in log, the actual values should be entered (the min and max of the entire image are output), since the log scale could well have an additive offset.
8. **Get pixels :** This is the same the normal **gp** option in the main menu whereby the x and y coordinates and the intensity will be output for eveyr pixel which is clicked on. It ends when you click outside the image. Note that this option can now be used inside **di** after zooming in.
9. **Toggle log/linear :** This toggles the greyscale/colourscheme between linear and logarithmic (base 10).
10. **Plot gaussians :** Overplots gaussians on the image. The input is a filename and the gaussians are assumed to be in *srldir*/filename.gaul, which is the standard BDSM gaussian list output format. The pixel positions in the .gaul file are plotted and hence the gaussian file has to correspond to the geometry of the image. An external catalogue can be converted to the .gaul file for a given image using the task 'st2g' in the 'sm' sub-menu, and the overlaid. Can be plotted on a zoomed-in image.
11. **Plot islands :** If BDSM has been run on the image, then the island boundaries, stored in the filename.islandlist file will be overlaid (also on a zoomed-in image) and the island number is also written on the island.
12. **Name stars :** A star-filename and a solution-filename needs to be input where *srldir*/starfilename.star is an AIPS-style star file [1]. There should also exist a BDSM-style gaussian file called *srldir*/starfilename.solu filename.gaul which can be created from the starfile using 'st2g' in the 'sm' sub-menu. The task then plots the stars with their names on the image. The advantage in this sub-task as opposed to using the 'Plots associatelist' option is that one can use an external catalogue which is not yet associated with the BDSM gaussian list.
13. **Add coordinate grid :** Overplots the coordinate grid, using the FITS header information written out to *srldir*/filename.solutionname.header by BDSM. If an image is read in using 'noise' instead (with 'rf') then this .header file is written out as well.

[1] with the format i2,1x,i2,1x,f6.3,1x,a1,i2,1x,i2,1x,f6.3,1x,f9.4,1x,f9.4,1x,f7.2,1x,i2,1x,f13.7,1x,a5

14. **Plot stars :** Overplots stars from an ascii file whose first two columns are the x and y coordinates. This does not check astrometry and does not care about the actual coordinates, unlike other tasks.
15. **Plots associatelist :** This opens up a new sub-menu, and is used to plot information from an associate list. A BDSM-style gaussian list can be associated with another using the option 'asrl' in the 'sm' sub-menu. This creates .assf, .asrl and .pasf files which are used by this task. The input is the name of the file, to be given without the path and a new sub-menu is opened up. The filename is of the format masterfile.soln1_slavefile.soln2 where slavefile.soln1.gaul is the name of the gaussian list associated against the masterfile.soln2.gaul. Zooming and greyscale status as well as all overlaid graphics are preserved.

    - **Circle associations :** Draws a circle encompassing the associated sources.
    - **Posn shift arrows :** Arrows are drawn from the position of a source in the master file towards the position of the associated sources in the slave file (and exaggerated in length by a factor of three).
    - **Print flux ratios :** The ratio of the peak flux of the master and slave associated sources is printed next to the source.
    - **Plot flux ratios :** Cirles are drawn such that the radius of the circle is linearly proportional to the flux ratio. Whether the circle is to be drawn centered at the positon of the master or the slave source is to be input.
    - **Print master name :** The masterfile name is input and the name of the associated source in the masterfile.star is printed.
    - **Print slave name :** The slavefile name is input and the name of the associated source in the masterfile.star is printed.
    - **Clear graphics :** Clears all graphics except the image.
    - **Quit :** Quits this sub-menu.

16. **Quit :** Quits the displayimage sub-menu.

3.5.1 Simple image operations

- *cc* : **Creates an image with a constant value.** Inputs are the name of the image (which is stored as name.img), size in X and Y directions and the constant intensity value (assumed to be Jy).

- *pp* : **Changes intensity values of pixels in an existing image.** Inputs are the name of the input image and the output image (without the *.img*, which will not be mentioned hereafter) and the word 'add' or 'put'. 'Add' implies that the intensity value input later is added to the value of that pixel and 'put' replaces the value. The image is then displayed. The pixel to be modified is then clicked on with the cursor following which the intensity value to be added/replaced is input (in Jy). Click outside the image to quit the task. The input and output image names can be the same.

- *gp* : **Outputs the intensity at a pixel.** The inputs are the image name and 'tv'/'wr' for clicking on the pixel on display or typing in the pixel coordinates. If the former, clicking outside the image exits the task and for the latter, (0,0) exits the task.

- *tr* : **Transposes an image.**

- *dt* : **Displays two images side by side** and also the difference and the ratio image, each with their own greyscale (which cant be changed, for the moment).

- *ci* : **Combines two images.** Input is the name of the two images, the name of the output image and either of 'add', 'sub', 'mul' and 'div'. Note that currently blanked pixels arent taken care of in this task.

- *ai* : **Adds multiple images.** Adds a number of images, all of which are assumed to have the same astrometry. The user is provided with a file of all the images or images which have a particular user-defined string and is asked to edit the list till all the images remaining in the list all have the same size. The user also inputs whether the images are to be added with all weights being equal, or given as the inverse square of the (5σ clipped) rms noise or taken from a user-specified file.

- *nb* : **n-bit an image.** First the image name is input. Next, the number of bits is entered. Next the user is prompted for the lowest integer level (that is, the lowest intensity range is represented as 0 or 1 etc - this is to avoid a large number of zeroes in the image). Next the values for the n-bit levels are input and the output image is written to inputname.nbit.

- *is* : **Computes image statistics.** Input is the image name and the value of κ. Then the image rms and mean and also the κ-clipped rms and mean and output along with the minimum and maximum intensities and their pixel positions. The histogram of the image within κ-σ of the mean is also plotted. Next one can repeat the same for a part of the image which is to be marked with the cursor. Note that, while marking with the cursor, the option to define an irregular area is not fully debugged. It loops back to ask for more regions.

- *io* : **Defines a function on an image.** This could either be a polynomial function or the absolute value function. The order of the polynomial is input, followed by the coefficients and this polynomial function of the intensity of each pixel is computed for the output image.

- *cl* : **Clips an image.** The minimum and maximum intensities are output along with a plot of the histogram (within sensible limits). Inputs then are range to be clipped and the value the clipped pixels are to have, followed by whether the clipping is to be done inside or outside this range.

- *mo* : **Moment analysis.** Calculates zeroeth, first and second moments of an input image for all pixels greater than nσ (n=3 currently) and the nσ masked image is written out as well. Should be extended to read in a mask file from user in the future.

- *su* : **Creates a sub-image** of an input image. Inputs are the input image name, output image name and 'tv'/'wr'. The former asks the user to draw the box on the image by marking the diagonal vertices. It then writes out the blc, trc on the screen and if one does not enter new blc, trc then the current ones are implemented. These can be changed at this step. If 'wr' is chosen then blc, trc is typed in by the user.

- *sl* : **Creates a slice plot.** User inputs the image name and whether a horizontal or vertical slice is desired. Then the slice is plotted through the pixel clicked. Clicking outside the image quits the task.

- *rb* : **Revalue blanked pixels.** Inputs are the current value of blanked pixel, the new value desired and the tolerance for the current value and an output is written with the new value of blanked pixels if there are any.

- *gc* : **Get sky coordinates.** The name of the fits file is input. The FITS keywords are then read in and the user is prompted for the pixel coordinates. The sky coordinates are then output. An input of (0,0) terminates the task.

3.5.2 Advanced image operations

- *cn* : **Creates a noise image.** Inputs are the name of the image to be created, the size in X and Y direction and the 1-sigma value in Jy of the image. Note that each pixel will be independent. The image is displayed on the screen.

- *cg* : **Convolves an image.** Inputs are the input image name, output image name and 'gaus'/'boxf' depending on convolution with gaussian kernel or boxcar smoothing. For the former, the FWHM of major and minor axes and the position angle are input in pixel units and the user can also specify whether the total flux or intensity will be conserved in the output image. For the latter option, the user specifies the width in X and Y directions of the smoothing box. Finally the histogram is plotted with the expected value of distribution overplotted along with the residue.

- *cs* : **Create sources.** This is a simple source simulation task inside 'noise'. A much better option for simulation is the package 'simulation' accessible through the same Makefile. The list of inputs are self-explanatory. The noise in the final convolved image is 1 Jy/beam. The outputs are the final image, the noise image and the list of sources added as a BDSM-style .gaul file which can be read by other tasks. The source intensities are randomly generated using the observed number counts from Katgert et al.

- *fg* : **Fit gaussian.** Fits a single gaussian to the entire image. The initial guess for the peak pixel position and the width is input by the user, so also is the cutoff used for masking the image in units of the rms. The fitted parameters are written out and the input image, residual, model and the ratio images are displayed.

- *dd* : **Detect discontinuities.** Computes an output image which detects discontinuities in the input image using the 3X3 matrix (-1,-1,-1,-1,-8,-1,-1,-1,-1). Can be easily extended to similiar alrogithms.

- *cm* : **Create modulations.** Creates an image with (currently) three sinusoids, which can be used as a noise image with large scale modulations (see '*mi*') to add to a source image. The amplitude of the waves is input by the user, so also is the number of wavelengths to be fit in the image. The angle of the waves is randomly generated. The image is displayed.

- *rf* : **Read in a fits image.** The full name of the fits file needs to be input, which is assumed to be in *fitsdir*. This is converted to a *.img* file and the fits headers are written to filename.header, both in *scratch*.

- *mi* : **Add sin waves to an image.** An image made using the task 'cm' can be added to an input image (after scaling by the ratios of the rms of the two images).

- *mf* : **Median filters an image.** An input image is median filtered with an user defined boxsize and the image of the median is written out. The input image and median image along with their histograms are plotted.

- *mn* : **Make a background noise map.** The input image and the names of the mean image, rms image and median image are typed in. These quantities are calculated in areas of size boxsize and these boxes are spaced stepsize apart. The edge effects are still not taken care

of totally satisfactorily. If stepsize is greater than unity, a linear or cubic spline interpolation can be used.

- *ac* : **Autocorrelation image** of an input image whose size is a power of 2. A gaussian can also be fit to the autocorrelation image.

- *sf* : **Structure function of an image.** The structure function of an image of any order can be calculated. The structure function image is plotted along with azimuthal profiles through the centre.

- *ft* : **Fourier transform of an image.** The fourier transform of an image is performed. If the image size is not a power of 2, it is padded with zeroes. The real part, imaginary part and the absolute value images are written out. The order of packing of the transform images is that of numerical recipes.

- *ps* : **Populate sources.** An image is made with just point sources, populated by Katgert et al. number counts, which can be used for confusion calculations. No convolution or addition of noise is performed. The sources put in are stored in a binary file. Source centres are assumed to be pixel centres. Use the task *cs* or the package simulation for a more sophisticated algorithm.

- *sc* : **n-σ cut.** If a pixel is less than an user-specified n times $\kappa(=3)$-σ clipped rms then it is made zero.

- *pb* : **Primary beam correction.** For an input image, user specifies the telescope, operating frequency, pixel scale and pointing centre. The VLA, WSRT, LOFAR and GMRT beams are known by the program. If the telescope is specified as USER, then the beam can be specified either as a gaussian (calculated from the diameter, which should therefore be the effective diameter) or a standard VLA/GMRT style polynomial (with four terms). The corresponding primary beam corrected image (multipled or divided) is written out.

3.5.3 Sub-menus

A number of sub-menus can be opened from *noise* which consist of a set of tasks for a specific purpose and are described below.

- *sm* : **Source Measurement (BDSM).** This opens up a menu which does two things : (1) all the steps of the automated command-line f-BDSM can be run individually and (2) two source lists can be associated and these can be plotted and analysed. The *sm* menu has the following tasks.

  First the BDSM tasks are listed below. Detailed descriptions of these will be found in the manual for f-BDSM.

  1. *list* : **Lists all images** with extension *.img*.
  2. *expe* : **Experiment.** Task for experimenting with preliminary algorithms.
  3. *quit* : **Quits** the *sm* sub-menu.
  4. *prep* : **Preprocess.** Preprocesses the image to calculate statistics as well as sensible parameters for the successive source extraction steps. These are written to *imagename.solutionname.bst* and *imagename.solnname.bparms* respectively.
  5. *rmsd* : **Background noise image.** Calculates the background noise image given a boxsize and stepsize.

6. *form* : **Form islands.** Forms islands of contiguous emission. .islandlist file.
7. *make* : **Makes island file.** Internal format stuff, to write a .islandlist and an ascii .islandlist.asc file.
8. *edit* : **Edit island file.** Can edit the .islandlist file using various criteria. Currently useful is by the minimum number of pixels per island (which is minpix_isl in the paradefine file for the automated f-BDSM.
9. *fiti* : **Fit islands.** Fits all islands with multiple gaussians and writes a .gaul file.
10. *resi* : **Makes a residual image** by subtracting all the fitted gaussians.
11. *plot* : **Plot sources/islands.** Overplots islands and gaussians on the image (which can also be done now using '*di*' in the main menu.

Next the source list association tasks :

1. *asrl* : **Associate source lists.** Inputs are the names of the master and slave .gaul files (without the extension). The search radius is asked for, and the (singly/multiply) associated and unassociated sources are all listed in human readable format in the masterfilename_slavefilename.assf in *scratch*.
2. *pasl* : **Plots associated list.** All quantities in the two .gaul files which have been associated can be plotted against each other. These plots can be zoomed into, and saved as well. If a device is specified as a file, note that the full path needs to be given, else it will be written in the current directory.
3. *csrl* : **Combine source lists.** Still being coded.
4. *aasl* : **Analyse associated list.** The fluxes, positions and sizes of all the associated sources in the two .gaul files are now analysed and the mean and rms are fit with standard functions, whose coefficients are written out.
5. *st2g* : **Converts star file to .gaul and .srl.** Useful to convert an AIPS-style star file (whose format is given in footnote 1) of an external catalogue into a .gaul and .srl file for overplotting and associating.

- *sh* : **Shapelets.** This opens up a seperate menu of tasks dealing with shapelets. Currently only cartesian shapelet decomposition is performed. Polar representations will be included later if needed.

1. *shap* : **Decomposes an image into shapelets.** Asks user for name of image file, followed by the shapelet scale β and n_{max}. It then calculates the best value of the centre which is output on the screen. The user can then enter these values again or a different pair or values. The display then shows the original image, the shapelet reconstruction (*imagename*.recons.img), the residual, and the shapelet coefficient matrix (coeff.img). Two other images are written as well, c1.img and c2.img, the images of the c_{12} and c_{21} coefficients assuming each pixel was the centre, used to find the optimal centre.
2. *varb* : **Find optimal β.** Task to calculate residual rms of an image as a function of β. The values of n_{max} and initial guess of β are input from user. The centre is computed and the user has a chance to change its value, as before. Then, for a range of values of β, the image is shapelet-decomposed, and the rms of the residual image is calculated and the value of β which minimises the residual rms is output.
3. *sims* : **Decompose simulated non-gaussian images.** This task simulates a number of images with non-gaussianity determined by inputs read from the file shapepara (which can be changed by the user). These images are then decomposed into shapelets and also fit by a gaussian.

4. *quit* : **Quits** the *sh* sub-menu.

- *ta* : **Test algorithms.** This is a set of test programs which are mainly for use by the author and will not be described further. However, its use is self-evident.

Chapter 4

Optimized Radio Global Sky Model

4.1 Introduction

This program (*or_gsm*) aims to construct a Global Sky Model (GSM) from a collection of sky catalogues, assumed to be at similar resolutions. A GSM is a model of the sky at a given resolution and frequency, in terms of discrete sources. This model is made by associating sources from the input catalogues. These GSM sources can be specified by a specific decomposition, e.g., gaussians or shapelets. Source properties are modelled as polynomial functions of frequency, though currently this option is only implemented for the total flux. Frequency spectra for each source are made available as a model as well as plots. A predicted GSM can be compared with an external catalogue as well.

4.2 Installation and Usage

or_gsm makes use of subroutines inside *noise* and can be compiled by the 'make or_gsm' command. This uses pgplot, cfitsio and wcslib like *noise* does. Running the script 'makeparadefine' will output the file 'gsmparadefine' which may need to be modified by the user. The user also then needs to run the script 'makegsmparadefine' which will set up the list of catalogues which the GSM will be constructed from. Following this, the executable *or_gsm* can be run.

4.3 Nature of inputs

The file 'gsmparadefine', written out by the script 'makeparadefine' has, apart from the standard directory pathnames, the following variables.

- gsmsolnname - identifying solution name for the *or_gsm* run.
- plotspectra - 'false'/'true', determines whether the spectra of all associated sources are plotted as postscript files or not.
- makepdf - 'false'/'true', determines whether a pdf file is made by putting together all the above spectra in one latex file. This takes time to run.
- spindef - the default spectral index to use to construct the model for sources which occur only in one catalogue.
- spinflux - 'total'/'peak', determines whether the spectral indices are calculated using the total flux or peak flux densities of sources.

The input catalogues have to be in the format of the BDSM .gaul files. Hence any image processed by BDSM can be used. If an image is not available, then an external catalogue should be first converted to the standard AIPS-format STAR file and then the task *st2g* in the sub-menu *sm* of *noise* should be used to construct BDSM format .gaul, .srl and .snum files.

When the script 'makegsmparadefine' is run, the user is prompted 'Use the (f)ile gsm_fits_list or create (n)ew one ? ' which has to be answered by f/n. *scratch*/gsm_fits_list is the file which has the list of catalogues to be input. If a new file is desired, then a file is opened with a list of all .gaul files (without the BDSM solnname extension) which has to be edited by hand (default editor is gvim) and the file closed. Next, after the user decided to proceed by pressing any key, another file is opened for the user to edit. The first field is the same as that of the previous file, the second field has the BDSM solnnames associated with the image (thus allowing multiple BDSM runs to be used) and the third field is either 0/1. If the entry is unity, then that catalogue is used to fit for spectral indices to model the radio spectrum of the associated source. If not, then that catalogue is not used to fit the spectral indices, but is used to make associations and the flux values are also output as well as plotted.

After the file is closed and a key is pressed to proceed, the user is asked to 'Use the (f)ile gsm_freqs or create (n)ew one ? ', which is answered as y/n. If a new file is desired, one is opened for editing, which will have a list of all the frequencies in MHz at which a GSM is desired to be predicted. After closing this file and proceeding, another file is opened for editing, which has the list of frequencies for prediction, a line saying END, followed by default values of spectral index parameters used for flagging a source, which can be changed by the user. These parameters relate to the frequency models currently used, which are :

$$\begin{aligned} \log(I) &= \log(I_{00}) + \alpha_{00} \times \log(\frac{\nu}{\nu_0}) \\ \log(I) &= \log(I_{10}) + \alpha_{10} \times \log(\frac{\nu}{\nu_0}) + \alpha_{11} \times \log^2(\frac{\nu}{\nu_0}) \end{aligned}$$

Now the executable *or_gsm* can be run.

4.4 What does *or_gsm* do

1. **Associates every pair.** Every pair of catalogues in the file *scratch/gsm_fits_list* are associated with each other. This can be used to identify catalogues with severe problems. The search radius is taken as the geometric mean of the median sizes of catalogue sources. Basic statistics like the mean, rms and median of the positon offsets, peak flux ratio and spectral index, for every pair of catalogues, are written out to a file.

2. **Associates all catalogues.** Next, all the input catalogues are associated to create a master associated-source list. The default is that only sources with code 'C' or 'S' in the .srl files (see BDSM chapter) are taken into account. Extended sources fit with multiple gaussians will be dealt with in the next release. The search radius for a pair of sources for a pair of catalogues is the minimum of fwhms of the sources in the direction of the other. Note that all 'C' and 'S' sources are included, including those with no counterparts in any other catalogue.

3. **Calculate spectral index for all associated sources.** Based on the flux density measurements at various frequencies corresponding to the input catalogues, the spectral index is calculated for each associated source. Only those catalogues whose third field in the file

scratch/gsm_fits_list is unity are included in the fitting procedure. If a source has only one frequency measurement, then the value of *spindef* in the file 'gsmparadefine' is taken. Else, I_{00} and α_{00} are calculated based on the above equation. If the number of frequency measurements exceeds two, then I_{10}, α_{10} and α_{11} are also calculated. The centroid position is also calculated for each associated source.

4. **Quality control.** The quantities calculated in the previous subroutine are plotted against one another as postscript files, and the statistics are also output on the plots for study. A file *gsm_spins_gsmsolnname* is written which has the clipped medians of the spectral indices α_{00} between every pair of catalogues, for both peak and total fluxes. This is useful to investigate curvature or whether a particular catalogue has flux calibration problems. These numbers are also conveniently plotted as *gsmsolnname_gsm_flux_allspin.png* where *flux* is either *peak* or *total*.

5. **Plot spectra of all associations.** If *plotspectra* in 'gsmparadefine' is true, then the spectra of all asosciated sources are plotted in *plotdir/gsmsolnname/* as individual files.

6. **Compute fluxes at frequencies not fit.** Using the spectral index parameters fit earlier, the flux density of each associated source is calculated at the frequencies of the catalogues which were not used for the fitting, viz., those with a zero in the third field in *scratch/gsm_fits_list*.

7. **Write the GSM.** The GSM is now written out in two files, i.e., *gsm_table_gsmsolnname* and *gsm_calcf_gsmsolnname*. The former has, for each associated source, the input flux from each catalogue (marked by its frequency), and the spectral index parameters with their errors. The latter file has the fluxes for each associated source including the calculated fluxes corresponding to the (frequencies of) catalogues which were not used for fitting. A file *gsm_posns_gsmsolnname* is also written which has the positions and their errors of all associations.

8. **Plot for unfitted frequencies.** The fluxes of every associated source have already been calculated for every catalogue not used for fitting and the ratio of the calculated to the catalogue fluxes are plotted as a function of RA, Dec, calculated flux, catalogue flux, distance from image centre, distance to the centroid of associated source etc, as seperate files. These files are in *plotdir/gsmsolnname/gsm_gsmsolnname_freq_cataloguename_fluxrat_quantity.png*. Also, a very useful file called *plotdir/gsmsolnname/gsm_cataloguename.gsm.fluxratio.ps* is plotted, which is the image of the unfitted catalogue (if it exists) with circles overplotted on each associated source, the size of the circle proportional to the ratio of the calculated to the catalogue flux.

9. **Write the GSM at input frequencies.** The GSM is calculated for the frequencies in the file *scratch/gsm_freqs* is calculated using the fitted spectral index parameters and is written as *scratch/gsm_calc_freqnumber*. The GSM written here has the RA, Dec, flux and their errors for each associated source.

4.5 Output products

The following are the (ascii) output products of *or_gsm* in *scratch/* :

- gsm_calcf_gsmsolnname : For each associated source, this has the flux densities from each input catalogue as well as the calculated flux densities at the frequencies not fitted. The flux density is the peak or total depending on *spinflux* in 'gsmparadefine'.

- gsm_table_gsmsolnname : For each associated source, this has the flux densities from each input catalogue as well as the fitted spectral index parameters and their errors.
- gsm_posns_gsmsolnname : For each associated source, this has the positions from each input catalogue as well as their errors.
- gsm_spins_gsmsolnname : Has the median of the spectral indices (α_{00}) of all associated sources for every pair of catalogues, for both peak and total fluxes.
- gsm_gsmsolnname_calc_fn : The GSM at the frequencies requested in gsm_freqs, which include the RA, Dec, flux and their errors, for frequency n.
- gsm_workf_gsmsolnname : Internal work file.

The following are the plots generated by *or_gsm* in *plotdir/gsmsolnname/* :

- gsm_total_allspin.png and gsm_peak_allspin.png : The median of spectral indices for total and peak fluxes between every pair of catalogues, output in gsm_spins_gsmsolnname, are plotted for ease, scaled to the highest frequency.
- gsm_gsmsolnname_gsmqc_qty1_qty2.png : Plots various fitted spectral index parameters (qty1 and qty2) against each other for all associated sources. The mean, rms and median of the x and y coordinate values are also listed in the plots.
- gsm_gsmsolnname_freqMHz_catimagename_catbdsmsolnname_fluxrat_qty.png : The ratio of calculated flux to catalogue flux for each catalogue which was not in fitting is plotted against quantities (qty). The catalogue is *catimagename* with the BDSM id of *catsolnname* and the freq is the the catalogue frequency in MHz. The various quantities are RA, Dec, calculated flux, observed flux, distance to centre of image and distance to centroid of associated source.
- gsm_catimagename.catbdsmsolnname.gsm.fluxratio.ps : This is the image of unfitted catalogue, if available, which is *catimagename* with BDSM id as *catbdsmsolnname*, with the associated sources overplotted as circles. The size of the circles is proportional to the ratio of calculated to observed flux.

4.6 Future capabilities

- Charactertise GSM sources by shapelets as well.
- Include extended sources with code 'M' properly.

Chapter 5

Analyse Islands

5.1 Introduction

This is a standalone program which will later be incorporated into ANAAMIKA and is useful for inspecting the results of BDSM for an individual island.

5.2 Installation and Usage

The command 'make analyse_island' in the ANAAMIKA directory will create the executable to be run. Running the standard script *makeparadefine* will create the file aiparadefine which then needs to be edited. Apart from the directory paths,

- the variable $fitsname$ stores the name of the fits file,
- *solnname* is the BDSM id,
- *gausshap* is the same as described for running BDSM and
- *saveplots* is 'y' or 'n', depending on whether the user wants the plots to be saved as .ps files or displayed on the screen using pgplot.

Note that this program assumes, obviously, that BDSM has already been run, with those indices of *gausshap* which are unity in *aiparadefine* also being unity in *paradefine*.

When the executable is run, the program quits is BDSM was not run properly. If it was, then the user is prompted for the isladn number. Entering zero quits the program and the directory of the output files is written out on the screen if *saveplots* was 'y'.

5.3 Products

Currently, only the shapelet products of an island are displayed/saved. For each shapelet, four FITS files are written for *saveplots*='y', namely

- $fitsname.solnname.num.subim.FITS$: This is the cutout image of the island, using the blc, trc in the .gaul file
- $fitsname.solnname.num.shap.FITS$: This is the shapelet reconstructed image of the island
- $fitsname.solnname.num.resid.shape.FITS$: This is the residual after subtracting the shapelet reconstructed image
- $fitsname.solnname.num.shapcoef.FITS$: This is a FITS binary table with the shapelet coefficients. The values of β, n_{max}, x and y coordinates of the shapelet centre as well the sky coordinates of the bottom left corner are written as FITS keywords (as described in the BDSM chapter)

where *num* is the island number. If *saveplots*='n', then these four are plotted on a single display, with the shapelet coefficient matrix plotted as an image as well.

5.4 Future capabilities

The gaussian analysis of islands will also be included in the next version. There will also be an option to redo the gaussian or shapelet analysis of the island with user defined input parameters.

Chapter 6

Quality Control

6.1 Introduction

This is a prototype of a Quality Control package, and is strung together from bits and pieces in the package ANAAMIKA to form an automated command-line software.

Currently, this package does the following :
(1) Calculates image statistics for every channel,
(2) Creates an image with the BDSM-detected gaussians overlaid,
(3) Associates the BDSM-gaussian list with an user-specified list of other gaussian lists (which could be created from other images through BDSM or imported from external catalogues) and
(4) Plots gaussian parameters of the BDSM-gaussian list against those provided by the user.

6.2 Contents and installation

The executable is called 'qualitycontrol'. The makefile can be used to create the exectuable. The requirements are the same as for ANAAMIKA .

The source finding software 'bdsm' needs to have been run on the input fits file before running 'qualitycontrol' (this reverse order of things will only be temporary) and checks if all the output files of 'bdsm' are present or not.

6.3 Usage: makeqcparadefine

First, the script makeparadefine is run, which creates the file paradefine. Check the name of plotdir inside makeparadefine. This is where the output plots are stored. The name of the input fits file then needs to be changes in the paradefine file. Next the script makeqcparadefine is run.

6.3.1 qc_asrl_list

Makeqcparadefine asks the user whether ze wants to use an existing version of the file qc_asrl_list or create a new one. This has a list of all the .gaul files to associate the source list of the input image with. If a new one is desired, this file is created and opened using gvim (change to your favourite editor by specifying it inside makeqcparadefine) and can be edited. Each row in the file has the file name and a number and a letter. The number signifies the association radius in arcseconds. The default is zero and is taken to imply half the beam size. The letter is either m or s and tells the software whether that source list is the master or the slave list (i.e., whether the bdsm sources are associated with this list or vice versa). When the user has finished editing the file, any key can be pressed to proceed.

If the user desires to import an external catalogue, ze first needs to convert it to .gaul and .srl from an AIPS-style STAR file, before running makeqcparadefine. This can be done by using the 'st2g' option in the 'sm' submenu of ANAAMIKA , as described in the manual. Please read this section of the manual for the exact format of the star file.

6.3.2 qc_pasl_opts

Makeqcparadefine then asks the user whether ze wants to use an existing version of the file qc_pasl_opts or create a new one. If a new one is desired, this file is created and opened using gvim. This file has the list of x and y axes to plot against each other for each of the associations defined in the qc_asrl_list above. The format is pretty self-evident, a new file is opened with sensible defaults and the codes are always given at the bottom of the file.

When done, run qualitycontrol and the list of plots are written out in a file whose name is printed on the screen.

Chapter 7

Simulation

7.1 Introduction

This is a fairly simple program to simulate the radio sky at low frequencies. Only single and double sources are incorporated currently. The need for upgrading the simulation to include extended sources is not foreseen, though.

7.2 Installation and Usage

The command 'make simulation' will create the executable in the ANAAMIKA directory. This needs only cfitsio and doesnt not use pgplot or wcslib. The directories for output are taken from the file 'paradefine', which is created by the script 'makeparadefine'. When the executable 'simulation' is run, the following inputs are required.

1. **File id :** The imagename of the output images.
2. **Maximum baseline (km) :** This sets the sythesized beam size.
3. **Antenna diamater (m) :** This sets the field of view.
4. **Integration time/pointing (hr) :** Integration time per pointing, used for sensitivity calculations.
5. **Bandwidth (MHz) :** This is used for sensitivity calculations.
6. **No. of channels :** If more than unity, an image cube is calculated.
7. **Frequency (MHz) :** Frequency of observation.
8. **Survey area (sq deg) :** The primary beam area is output first in sq deg and the user is asked for the total survey area.
9. **Constant beam (4) or vary (1/2/3) :** Options 2 and 3 are not yet implemented. Option 4 assumes a constant psf (synthesized beam) across the entire field. Option 1 assumes a simple model for the spatial variation of std of the beam major (M) and minor (m) axes which is a sum of a constant, a sine function of the coordinate and a linear term, whose powers are randomly generated for each run.
10. **Array (LOFAR=1, GMRT=2) :** This is used for sensitivity calculations.
11. **Reference pixel (x1, x2) :** This is the value of CRVAL.
12. **RA, Dec at refpix (hh mm ss.s Sdd Sma Ssa.sa) :** This is the value of CRPIX.
13. **No. of facets in RA,Dec:** Number of facets to divide the total image into, in two dimensions. Each facet is written finally as a FITS image file.
14. **Overlap pixels :** The number of pixels overlap between adjacent facets.
15. **No. of pointings in RA,Dec :** The number of individual telescope pointings making up the entire image. The primary beam is applied to the entire image for each pointing, which are then added and then broken up into facets.

7.3 Properties of the sky

- The source counts are taken from the 1400 MHz counts of Katgert et al. 1988 and Richards 2000, as calculated by Rottgering. These are computed at the observing frequency using a spectral index of -0.7. These non-linear source counts are split into five parts, each of which are fit by a straight line in the log-log domain to facilitate computation of a population randomly drawn from this distribution.

- The positions of each source are chosen randomly. The source sizes are taken from Windhorst 1990, where the median size L_{med} in arcseconds is related to the 1.4 GHz flux density in mJy as 2.0 $S^{0.3}_{1.4GHz}$ and the individual sizes are randomly drawn from the size distribution given as $f(L) = exp(-\ln 2\ (L/L_{med})^{0.62})$. The P.A. of the source shape is random as well.

- An image of the source with the above properties is then constructed. The source can be either a point source or a double source. If the latter, then the flux between the two point source components is divided according to a gaussian distribution of mean 0.5 and std 0.2.

- Each source subimage, with the properties as described above, is convolved with the psf. This psf could either be constant or vary with position, as specified by the user.

- The sourcelist is written out in *srldir/imagename.srclist* and will be written in .gaul format in the next release.

- The convolved noise map is created, corresponding to the instrument sensitivity, and seperately for each channel.

- If the number of channels is more than one, then the source cube is created by assuming a spectral index which is gaussian distributed with a mean of -0.8 and dispersion of 0.2.

- If there is more than one pointing, then the primayr beam is applied to all the pointing centres in the image. If the number of facets is more than one, then this image is split into a number of facet images and written as FITS files.

7.4 Output products

The following are the final data products.

- $fitsdir/imagename.n.FITS$: These are the image cubes as FITS files, where n is the number of the facet.

- *srldir/imagename.srclist* : ascii file of the source list, to be make *noise* readable .gaul format.

The following are *noise* readable binary images in *scratch*

- *imagename.src.img* : This is the single plane image of sources.

- *imagename.src3D.img* : This is the image cube of the sources (if number of channels is more than one).

- *imagename.noi.img* : The simulated noise image, single channel.

- *imagename.src3D.pbeam.img* : This is the source image cube after applying primary beam correction depending on the number of pointings.

- *imagename.bmajor.img* : If psf is chosen to be variable, this is the image of the std of the beam major axis.
- *imagename.bminor.img* : If psf is chosen to be variable, this is the image of the std of the beam minor axis.

Chapter 8

Notes for a short demo for ANAAMIKA

8.1 Starting up

You need to install anaamika and successfully compile the modules noise, qualitycontrol, simulation, or_gsm, analyse_island, and bdsm. It would be best if you make sure you have installed stuff before the demo.

1. Download latest anaamika version from http://www.strw.leidenuniv.nl/ mohan/anaamika onto your desktop or laptop. Also download the documentation linked from there. Read the installation section in chapters 1, 2, 3, its pretty simple.

2. The most painful part is getting pgplot to work. Its no longer supported by GNU so you need an older f-compiler for it - ask David Jansen for the same version as he has installed on my machine. If you have a mac, ask Alexander Usov to help you fix it. Installing wcslib and cfitsio is straightforward on your laptops. You can get just link up to my versions of wcslib and cfitsion at /disks/strw4/mohan/ - modify your Makefile suitably.

3. If you are on a mac, you need to run the script linux2mac (modify it to reflect your own dir structure).

4. In the main directory 'anaamika', you need to 'make' noise, qualitycontrol, simulation, or_gsm and analyse_island. Modify the script makeparadefine to change the paths of the working directories scratch, plotdir, fitsdir and srldir (i usually make srldir the same as scratch and would advise the same).

5. You then need to go to the sub-dir bdsm-dir and run the script 'create' and then make bdsm.

6. 'unlimit', run the script makeparadefine in the two directories and you are set to go.

8.2 'noise' and 'bdsm'

8.2.1 Basic image fudging

First we will learn how to create fake images, modify them, do simple operations on them, We will also learn how to display images and admire how pretty they are. Text in bold is what the user inputs.

- **cn** : And the Earth was without form, and void ...
  We first create a 256^2 noise image called noise, with dispersion unity. Press return when

asked, and just click on the 'quit' option on the graphics menu for now.

Option − − − > **cn**
imagefilename xsize ysize sigma(Jy)
noise 256 256 1
Image of noise ... <RET>
Writing out /Users/mohan/scratch/scratch/noise.img

- **pp** : Let there be light: and there was light ...
We can now change the values of individual pixels, so lets create emission.

Option − − − > **pp**
imagefilename outputfilename add/put
noise putpix put
*Flux (Jy) at this pixel :***100**
Flux (Jy) at this pixel : **75**
Writing out /Users/mohan/scratch/scratch/putpix.img

- **cg** : And you see the light, that it is good: and you divide the light from the darkness ...
We can convolve the image now and make it look real.
Option − − − > **cg**
inputname outputname filter (gaus/boxf)
putpix image1 gaus
FWHM (bmaj, bmin, bpa) :
8 5 30
Preserve (t)otal flux or (i)ntensity **i**
Using gaussian image of size 37
At edge, condition is periodic
Image of convolved noise ... <RET>
Plot of fit <Ret>
computing (gaussian) image statistics ...
Peak = 0.678774E+04, Histo fitted RMS = 0.457525E+01 Centre = -0.193413E+00
Max and min are 21.9524478478815 & -4.27293188260362 sigma resp.
Writing out /Users/mohan/scratch/scratch/image1.img

- We can now see if the flux scale is correct.
Option − − − > **is**
image k-sigma
image1 5
Statistics over the entire image ...
Image std = 5.358E+03 mJy ; Mean = -4.631E+01 mJy
Total flux = -3.035E+03 Jy ; Area = 65536 pixels
5-sig clip std = 4.794E+03 mJy ; mean = -1.466E+02 mJy
Image maximum = 1.002E+02 Jy at 91, 138 ; snr = 20.9
Image minimum = -1.974E+01 Jy at 70, 67 ; snr = -4.1

Want statistics over a region (n) ? **n**

- We can get the intensities at specific pixels as well.
Option − − − > **gp**
imagefilename tv/wr

image1 wr
Enter coordinates (0,0 to quit) : **91 138**
Coordinates : 91 138 ; Flux density : 1.002445E+02 Jy/beam
Enter coordinates (0,0 to quit) : **10 100**
Coordinates : 10 100 ; Flux density : 3.934066E-01 Jy/beam
Enter coordinates (0,0 to quit) : **0 0**

- We can now median filter the image.
Option − − − > **mf**
imagefile opfile filtersize
image1 image2 17
Median filtering the data ...
Writing out /Users/mohan/scratch/scratch/image2.img
Image of median ... <Ret>

- And calculate the difference image.
Option − − − > **ci**
image1 image2 outputimage oper
(oper = add/sub/mul/div)
image1 image2 image3 sub
Writing out /Users/mohan/scratch/scratch/image3.img

- Lastly, lets display the image. The menus are self explanatory. For now, lets just play with zooming, colourscale and pixel intensity range.
Option − − − > **di**
imagefile
image1

- And quit.
Option *− − − >* **qn**

8.2.2 Intermediate image fudging

- First, read in a fits file. This will be written, as all images are, as a binary file *.img.
Option − − − > **rf**
FITSfilename
subim.fits
Reading /Users/mohan/scratch/fits/subim.fits
Size of FITS image is 1501 X 1385 X 1 X 1
Writing out /Users/mohan/scratch/scratch/subim.img
Now you can display subim.img using **di**.

- If the image has blanks and you dont want to deal with them right now, you can change the values using **rb**. Note that the magic value for blank inside anaamika is -999.0 and not IEEE NaN.

- Lets cutout a source from this image. We can use the tv and then change the coordinates later if we want.
Option − − − > **su**
inputimage outputimage tv/wr
subim cut tv
tvwin first, then correct blc,trc

Set the window ...
BLC is 772 824
TRC is 1226 1284
Enter new blc trc (y/n) ? **y**
Enter BLCs and then TRCs : **759 1194 790 1225**
Writing out /Users/mohan/scratch/scratch/cut.img

- Compute the moments of this image next.
  Option − − − > **mo**
  inputfile
  cut
  Writing out /Users/mohan/scratch/scratch/cut.mommask.img
  moment 0 = 1.49E-01 Jy
  moment 1 = (18.33, 16.42)
  moment 2 = 3.18, 3.45 pixels

- We can then compute its fourier transform using **ft** or its structure function or autocorrelation function using **sf** or **ac**.

- We can then write out cut.img as a fits file (PROBLEM ... doesnt inherit correct fits headers yet if done this way).
  Option − − − > **ff**
  imagefilename
  cut.img
  File is binary
  Conversion (b2a a2b b2f a2f) : b2f

- We can fit a single gaussian now to it as well, quickly. We can then look at input.img and resid.img.
  Option − − − > **fg**
  imagefile
  cut
  3-sig clip std = 1.39E-01 mJy ; mean = -1.69E-02 mJy
  Approximate FWHM in pixels : **5**
  Cutoff in Jy : **0.0005**
  (f)it all or (s)ymmetric gauss : **f**
  Fitted values are :
  Peak = 1.115E-02 Jy; Centre is (18.34, 16.45)
  Fitted size = 3.24 X 3.78 X 142.45
  Red. chi sq = 0.796388751500758 28.6699950540273 36

  Writing out /Users/mohan/scratch/scratch/input.img
  Writing out /Users/mohan/scratch/scratch/resid.img

- Now to create a more realistic fake image. You can use **cs** to create less realistic images but more useful for debugging BDSM.
  Option − − − > **ps**
  filename
  op
  Array : 1. LOFAR 2. LOFAR (VC) 3. GMRT 4. WSRT 5. VLA : **3**

Freq band (MHz) : 1. 150 2. 230 3. 325 4. 610 5.1400 : **3**
Integration time (hrs) : **1**
Bandwidth (MHz) : **2**
No. pixels/fwhm of clean beam : **3**
Image size (fw=2666 pix) : **128**
1 sigma is 8.49E-02 mJy
NOTE: Source peaks are assumed at pixel centres
NOTE: Extended sources not put in properly
Putting in 64992.7087295388 sources ...
64239.3658219403 srcs bet 6.992587267111725E-003 & 0.176784212118387
746.242493381458 srcs bet 0.176784212118387 & 4.46939822475392
6.73031994480235 srcs bet 4.46939822475392 & 112.993857364515
0.339737366387328 srcs bet 112.993857364515 & 2856.67057379364
Image std = 1.50E-02 mJy ; mean = 4.50E-03 mJy
5-sig clip std = 2.88E-03 mJy ; mean = 3.50E-03 mJy
Writing out /Users/mohan/scratch/scratch/op.img
We can now create a noise image **cn**, add it to op.img **ci** and convolve with the restoring beam **cg**.

- Now create an extended source and analyse it using the **sh** submenu - **shap** and **varb**.

8.2.3 Running BDSM

First go to the bdsm-dir directory, run the script 'create', run the script 'makeparadefine', and edit paradefine. You can now 'make bdsm', and run bdsm on the single file. There is also a batchprocess script for multiple files. The name of the output file is given at the end of the processing. For more details, look at BDSM chapter of the manual.

Look at the output files of BDSM yourself.

8.2.4 Advanced image fudging

Here we will inspect the results of BDSM, inside noise, and compare with external catalogues.

- Use testranbdsm to verify that all files have been written by BDSM. Use **li** inside noise to list all of these. Know thy files.

- Load the image using **di** and adjust the grey/colourscale. Overplot the islands. Next overplot the initial guess for gaussians. Clear the screen (zoom out) and overplot the fitted gaussians.

- Now we want to see if the image and BDSM results agree with whats known from other catalogues or images. So lets first run BDSM on cs1_rvw1.fits. Also on nvss_cs1 (the first is averaged by Reinout and the second was created by the same). We also have the 4crJ.star and snr.star which are AIPS style star files I created from the complete catalogues.

  1. First convert snr.star and 4crJ.star to the astrometry of cs1_rvwi.fits and write the source lists in the BDSM format. Go to the **sm** submenu of 'noise'.
     BDSM_Option $---> $ **st2g**
     name file.solnnname (name.star –¿ file.solnname.gaul
     snr cs1_rvwi.oldmonk
     And similarly for 4crJ.star as well.

2. Let us associate pairs of source lists now.
   BDSM_Option $- - - >$ **asrl**
   mastersourcelist secondsourcelist
   cs1_rvwi.oldmonk snr.s2g
   Input images are different (cs1_rvwi & snr)
   Want to continue (y)/n : **y**
   Median sizes : 1991.396 & 330.000
   Tolerance : **2000**
   Associating sourcelists cs1_rvwi.oldmonk (442 srcs) & snr.s2g (265 srcs).
   Writing out /Users/mohan/scratch/scratch/cs1_rvwi.oldmonk_snr.s2g.asrl
   Found 15 associations with cs1_rvwi.oldmonk
   Sources in cs1_rvwi.oldmonk with no associations is 427
   Sources in snr.s2g with no associations is 250
   Sources in cs1_rvwi.oldmonk with multiple associations is 0
   Sources in snr.s2g with multiple associations is 0

   Writing out /Users/mohan/scratch/scratch/cs1_rvwi.oldmonk_snr.s2g.assf
   Writing out /Users/mohan/scratch/scratch/cs1_rvwi.oldmonk_snr.s2g.pasf
   Writing out /Users/mohan/scratch/scratch/cs1_rvwi.oldmonk_snr.s2g.gasl
   Do the same for 4crJ.s2g as well.
3. Now you can plot one quantity against the other. Use the option **pasl**.
4. Quit BDSM submenu, display the image again and overplot the 4crJ.s2g.gaul and snr.s2g.gaul as well as the BDSM gaussians. Do the same on the residual image. Use **dt** to display the original and residual image as well.
5. Next, display the image again, click on 'Plots associatelist', enter the snr.s2g association, to open the associatelist submenu which you can now explore.
6. You can plot the 4crJ or snr stars which are associated inside the submenu above and also the entire catalogue in the main display menu.

- We can now run the executable analyse_island to inspect the result for each island. Right now, this is implemented for shapelets and not for gaussians. Use this to inspect shapelet decomposition in conjunction with the shapelet.c file.

8.3 OR_GSM

In this section, we learn how to create a GSM. First please read the ORGSM chapter of the manual.

8.4 Quality Control

Read the manual and just run it.

Chapter 9

Changes since previous version

- Feb 22 2008 : The FITS binary table of the source catalogue *.srl.FITS was corrupted. Fixed now.
- Feb 23 2008 : LONPOLE was explicitly defined as 180.0 degrees while calling WCS for cases when CRVAL for Declination was exactly 90 degrees.
- Apr 22 2008 : Added a AIPS-style TVSTAR option to 'noise' where an ascii file whose first two columns are x and y positions can be overplotted on an image using 'di' irrespective of astrometry.
- Apr 22 2008 : Fixed wcs_struct.f to handle non-zero CROTA values properly by giving suitable values of ALTLIN.
- Apr 22 2008 : Pixels which are outside the universe are now handled correctly even at the rms image forming as well as the island forming stage.
- Apr 22 2008 : A bug in WCSLIB4.3 was fixed by increasing value of WCSLEN parameter.
- Apr 24 2008 : A new file gsm_posns_gsmsolnname is created by or_gsm which stores all the positions and their errors of all associations.