SC/22

Starlink Project
Starlink Cookbook 22

H. A. L. Parsons, D. S. Berry, M. G. Rawlings, and S. F. Graves

2021 July 15

Copyright © 2021 East Asian Observatory

The POL-2 Data Reduction Cookbook 1.1

Abstract

This cookbook provides an introduction to POL-2 data reduction, using the Starlink facilities SMURF(the Sub-Millimetre User Reduction Facility) and in particular its command `pol2map`. This cookbook illustrates the various steps required to reduce the data, including an overview of the method. It also describes how to calibrate and display the data as images or vector maps.

Copyright © 2021 East Asian Observatory

Contents

List of Figures

Acronyms

CADC	Canadian Astronomy Data Centre
FCF	Flux Conversion Factor
FITS	Flexible Image Transport System
GAIA	Graphical Astronomy and Image Analysis tool
HWP	Half-Wave Plate
ITC	Integration Time Calculator
I	Total intensity
IP	Instrumental Polarisation
JCMT	James Clerk Maxwell Telescope
NDF	Extensible N-Dimensional Data Format
P	Percentage polarisation
PCA	Principal Component Analysis
$\mathbf{I}_p$	Polarised intensity
SCUBA-2	Submillimetre Common User Bolometer Array-2
SMURF	Sub-Millimetre User Reduction Facility
SNR	Signal-to-noise ratio
SUN	Starlink User Note
WCS	World Coordinate System

Chapter 1
Introduction

1.1 This cookbook

This guide is designed to instruct POL-2 users on the best ways to reduce and visualise their data using Starlink packages: SMURF[6], KAPPA[9], POLPACK[4], and GAIA[12].

This guide covers the following topics.

- Chapter 1 – Computer resources needed before getting started.
- Chapter 2 – A description of POL-2 and its observing modes.
- Chapter 3 – POL-2 Data Reduction - The Theory
- Chapter 4 – POL-2 Data Reduction - Running `pol2map`
- Chapter 5 – POL-2 Image Display
- Chapter 6 – POL-2 Advanced Data Reduction

Throughout this document, a percent sign (`%`) is used to represent the Unix shell prompt. What follows each `%` will be the text that you should type to initiate the described action.

1.2 Before you start: computing resources

Compared with SCUBA-2 observations, POL-2 observations are far less memory-intensive to reduce. POL-2 time-series data are down-sampled to 2 Hz as a part of the reduction process. Assuming a typical 35-minute POL-2 observation, the reduction requires 35 GB of memory (in comparison with SCUBA-2 maps that may require up to 96 GB of memory).

The main consideration for POL-2 reductions is processing power. PCA calculations in makemap can be lengthy, so fast processors with lots of cores are advised.

1.3 Before you start: software

This manual uses software from Starlink packages: SMURF [6], KAPPA [9], POLPACK[4], and GAIA [12]. Starlink software must be installed on your system, and Starlink aliases and environment variables must be defined before attempting to reduce any SCUBA-2 data (see Section 1.3.2).

1.3.1 Data formats

Data files for POL-2 are structurally the same as for SCUBA-2, and use the Starlink *N*-dimensional Data Format (NDF; [17]), a hierarchical format which allows additional data and metadata to be stored within a single file. KAPPA contains many commands for examining and manipulating NDF structures. The introductory sections of the KAPPA document (SUN/95) contain much useful information on the contents of an NDF structure and how to manipulate them.

A single NDF structure describes a single data array with associated meta-data. NDFs are usually stored within files of type `.sdf`. In most cases (but not all), a single `.sdf` file will contain just one top-level NDF structure, and the NDF can be referred to simply by giving the name of the file (with or without the `.sdf` suffix). In many cases, a top-level NDF containing JCMT data will contain other 'extension' NDFs buried inside them at a lower level. For instance, raw files contain a number of NDF components, which store observation-specific data necessary for subsequent processing. The contents of these (and other NDF) files may be listed with HDSTRACE. Each file holding raw JCMT data on disk is also known as a 'sub-scan'.

The main components of any NDF structure are:

- an array of numerical data (which may have up to seven dimensions—usually three for JCMT data);
- an array of variance values corresponding to the numerical data values;
- an array holding up to eight Boolean flags (known as 'quality flags') for each pixel;
- World Co-ordinate System information;
- history;
- data units; and
- other extensions items. These are defined by particular packages, but usually include a list of FITS-like headers together with provenance information that indicates how the NDF was created. Raw JCMT files also include extensions that define the state of the telescope and instrument at each time slice within the observation.

The Starlink CONVERT package contains commands fits2ndf and ndf2fits that allow interchange between FITS and NDF format.

1.3.2 Initialising Starlink

The commands and environment variables needed to start up the required Starlink packages (SMURF[6], KAPPA, *etc.*) must first be defined. For C shells (csh, tcsh), the commands are:

```
% setenv STARLINK_DIR <path to the starlink installation>
% source $STARLINK_DIR/etc/login
% source $STARLINK_DIR/etc/cshrc
```

before using any Starlink commands. For Bourne shells (sh, bash, zsh), the commands are as follows.

```
% export STARLINK_DIR=<path to the starlink installation>
% source $STARLINK_DIR/etc/profile
```

1.3.3 KAPPA and SMURF for data processing

The Starlink Sub-Millimetre User Reduction Facility package, or SMURF, contains the Dynamic Iterative Map-Maker, which will process SCUBA-2 time-series data into images (see **SUN/258**). KAPPA, meanwhile, is an application package comprising general-purpose commands mostly for manipulating and visualising NDF data (see **SUN/95**). Before starting any data reduction it is necessary to initiate both SMURF and KAPPA.

```
% smurf
% kappa
```

After entering the above commands, the help information for the two packages can be accessed by typing smurfhelp or kaphelp respectively in a terminal, or by using the showme facility to access the hypertext documentation. See Section 1.3.5 for more information.

Tip

The `.sdf` extension on file names need not be specified when running most Starlink commands (the exception is PICARD).

1.3.4 GAIA for viewing your images and vector maps

Images and vector maps can be displayed and analysed using GAIA (see **SUN/214**) – an interactive GUI-driven tool that incorporates facilities such as vector selection, vector binning, source detection, photometry and the ability to query and overlay on-line or local catalogue data.

```
% gaia map.sdf
```

Alternatively, the KAPPA package includes many visualisation commands that can be run from the shell command-line or incorporated easily into your own scripts—see Appendix “Classified KAPPA commands” in SUN/95.

1.3.5 How to get help

Help command	Description	Usage
showme	If the name of the Starlink document to be viewed is known, then showme can be used. When run, the default browser opens a new web page or tab displaying the hypertext version of the document.	`% showme sun95`
findme	findme searches Starlink documents for a keyword. When run, the default browser opens a new web page or tab listing the results.	`% findme kappa`
docfind	docfind searches the internal list files for keywords. It then searches the document titles. The results are displayed using the Unix more command.	`% docfind kappa`
Run (non-script) routines with prompts	Any routine that is not implemented by a script (i.e. all compiled commands written in C or Fortran) may be run with the `prompt` option after the command. This will prompt for every parameter available. If a further description of any parameter in needed, type `?` at the relevant prompt. Note that this option is not available for `pol2map` (because it is script-based).	`% makemap prompt` `% REF - Ref. NDF /!/> ?`
Google	A simple Google search such as "`starlink kappa fitslist`" will usually return links to the appropriate documents. However, the results may include links to out-of-date versions of the document hosted at non-Starlink sites. Users should always look for results in `"www.starlink.ac.uk/docs` (or `"www.starlink.ac.uk/devdocs` for the current development version of the document).	

Chapter 2
POL-2 Overview

2.1 The instrument

The POL-2 instrument is a linear polarimetry module for the Submillimetre Common User Bolometer Array-2 (SCUBA-2), a 10,000 bolometer camera on the JCMT [13] [2]. POL-2 in itself is not a detector—thus requiring SCUBA-2 and its detectors for operation. SCUBA-2 (and consequently POL-2) operates simultaneously at both 850 and 450 μm.

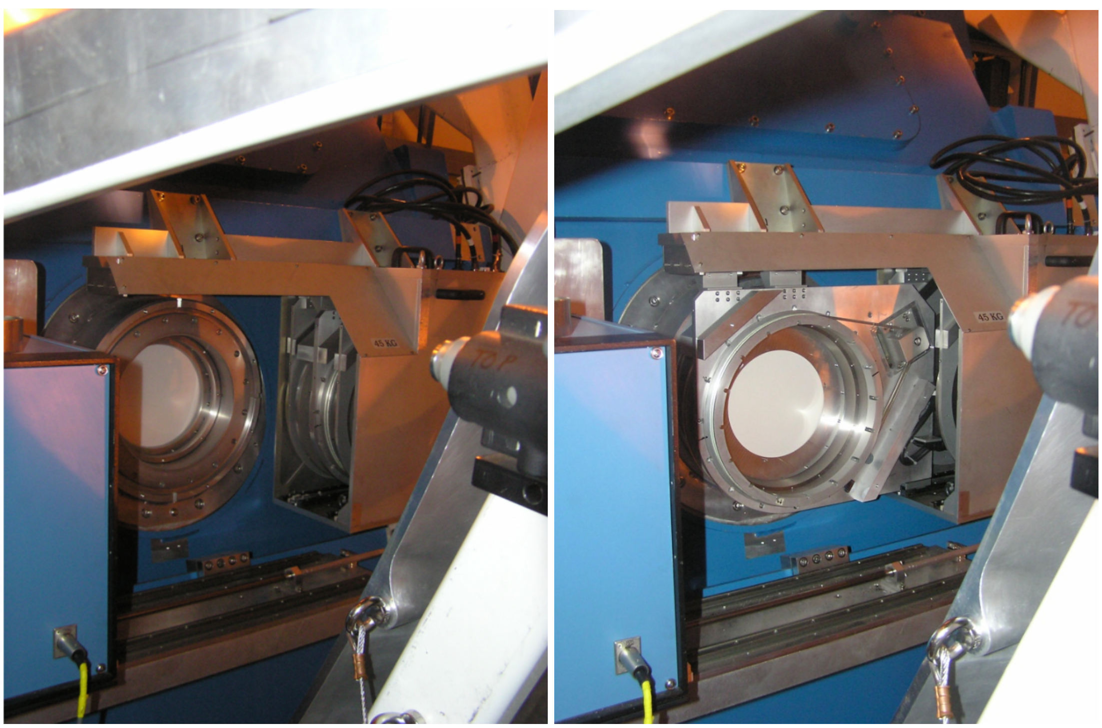


Figure 2.1: POL-2 mounted on the front of SCUBA-2. The left image shows the SCUBA-2 window. The right image shows the components of POL-2 inserted in front of the SCUBA-2 window: the calibrator grid, rotating half-wave-plate (HWP) and the analyser grid. The calibrator grid is only inserted for test purposes.

Polarisation

In polarimetric terms, light is conventionally described by the four Stokes parameters: I, Q, U, and V.

I is the total intensity; Q is the radiation linearly polarised in the direction parallel or perpendicular to the reference plane. U is the radiation linearly polarised in the directions 45° to the reference plane; and V is the circularly polarised radiation.

POL-2 is designed to characterise linear polarisation. The V parameter, consequently, is not discussed further, with subsequent focus being only on I, Q, and U.

The linear Polarised Intensity (I_p) and polarisation angle (θ) can be described as:

$$I_p = \sqrt{Q^2 + U^2} \tag{2.1}$$

$$\theta = 0.5 \arctan(U/Q) \tag{2.2}$$

with Q and U related to the polarisation angle and the polarised intensity by:

$$Q = I_p \cos(2\theta) \tag{2.3}$$

$$U = I_p \sin(2\theta) \tag{2.4}$$

where

$$Q = Q_m - I.ip_q \tag{2.5}$$

$$U = U_m - I.ip_u \tag{2.6}$$

where Q_m and U_m are the measured values of Q and U before correction for instrumental polarisation (IP) is applied. The IP correction is the product of the total intensity (I) and a factor (ip_q or ip_u) that varies with elevation but which is always smaller than 0.03.

How POL-2 works

POL-2 is located in front of the window to the SCUBA-2 instrument (as is seen in Figure 2.1), and covers the full field of view of SCUBA-2. The POL-2 polarimeter uses three optical components that cover the full field of SCUBA-2:

(1) a wire-grid polariser used as a calibrator (only included in the beam for test purposes),

(2) a Half-Wave Plate (HWP), and

(3) a second wire-grid polariser used as an analyser.

These components can be seen in Figure 2.1. A schematic of POL-2 is given in Figure 2.2.

Rotating the HWP rotates any linearly polarised component of incoming radiation. The HWP rotates this incoming linear polarisation with twice the speed of the HWP angle (δ) producing the *effective analyser* position (ϕ—as defined in the POLPACK documentation), such that:

$$\phi = 2\delta. \tag{2.7}$$

The rotating linearly polarised component is transmitted or reflected by the grid, causing a modulation in the transmitted intensity. The amplitude of the polarised component transmitted by the polariser is $\sim$cos(ϕ) while the power is $\sim\cos^2(\phi)$.

The radiation passing through the polarimeter is detected by SCUBA-2. The detected intensity (I_{detected}) is a combination of *both* the unpolarised intensity ($I_{\mathrm{unpolarised}}$) and the linearly polarised intensity (I_p)[1]. This detected intensity can be described by:

[1]The total intensity of the source, I, is $I_{\mathrm{unpolarised}} + I_p$.

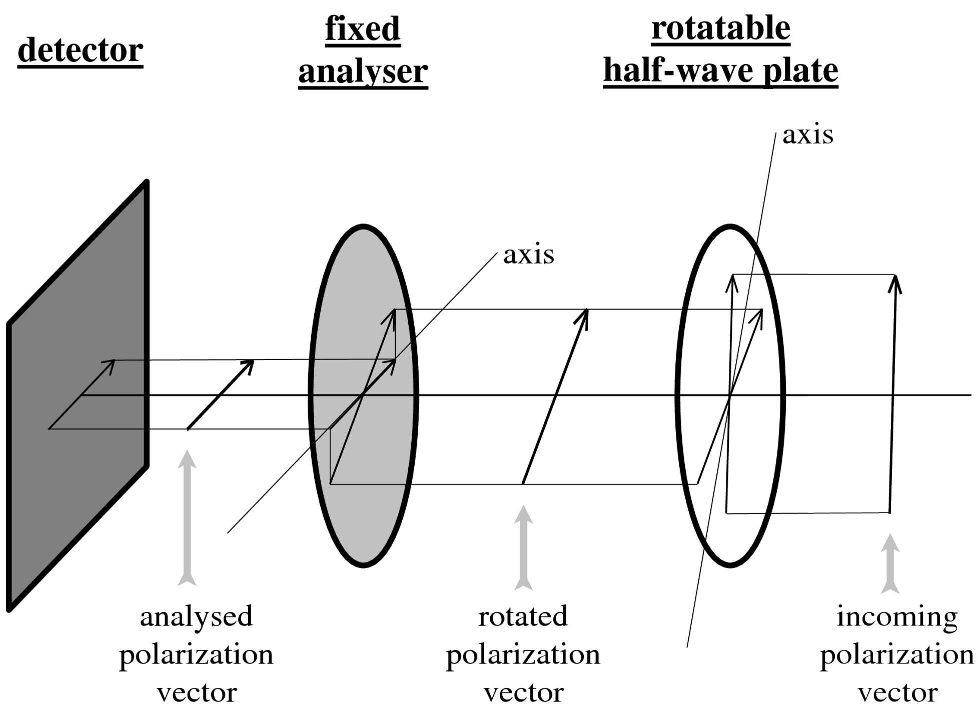


Figure 2.2: The main optical components in a typical single-beam imaging polarimeter such as POL-2 (taken from SUN/223).

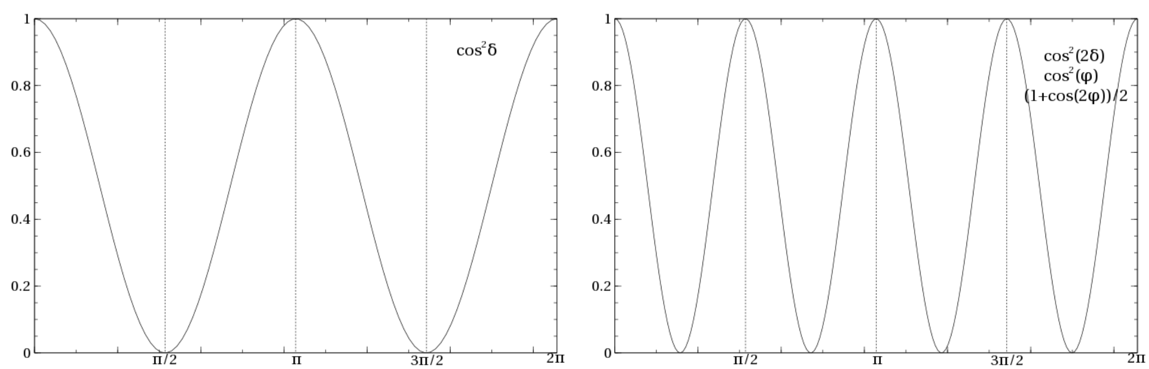


Figure 2.3: Left: If there was a single rotating analyser this would be the resulting curve of the power transmitted of the linearly polarised component. Right: With the HWP, the linearly polarised component is rotated at twice the speed. It may be useful to remind the reader of the trigonometric identity: $\cos^2 x = 0.5(1+\cos(2x))$

$$I_{\text{detected}} = \frac{I_{\text{unpolarised}}}{2} + I_{\text{p}} \cdot \left(\frac{1 + \cos(2\phi - 2\theta)}{2} \right) \tag{2.8}$$

with the above equation being in terms of the effective analyser angle, ϕ, and the angle of the polarisation (θ). This can also be expressed in terms of the the HWP angle (δ).

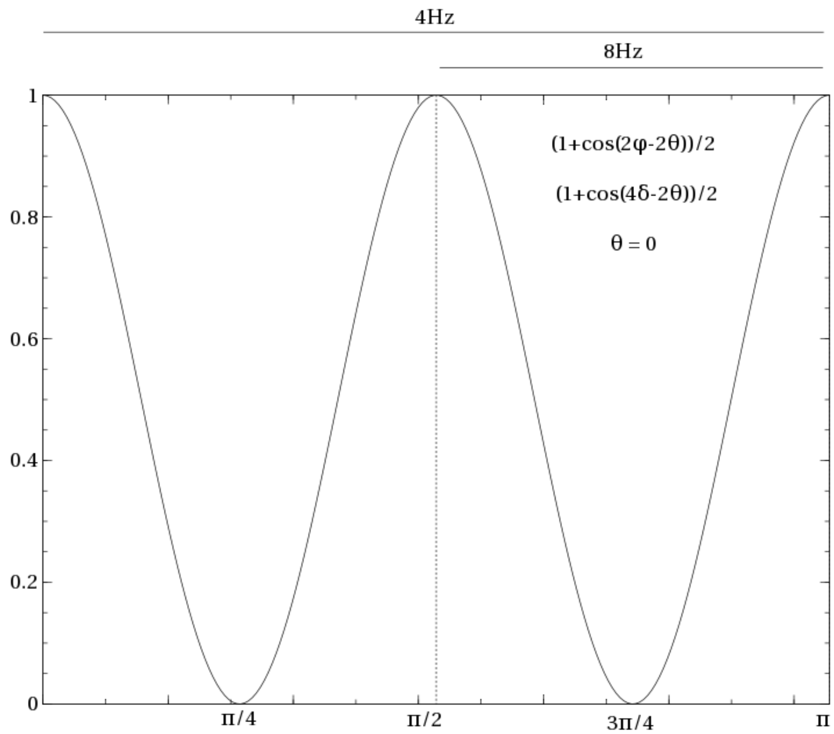


Figure 2.4: The incoming polarised radiation (with a polarised angle, θ, of zero) is attenuated by the HWP. The HWP rotates at 2Hz (through 2π) so we see the signal is modulated at 8Hz as the instrument scans at $8''/\mathrm{s}$.

$$I_{\text{detected}} = \frac{I_{\text{unpolarised}}}{2} + I_{\text{p}} \cdot \left(\frac{1 + \cos(4\delta - 2\theta)}{2} \right) \tag{2.9}$$

The Half-Wave Plate

As described in the POL-2 commissioning document, the HWP is constructed from five individual synthetic sapphire layers approximately 0.9 mm thick and 200 mm in diameter. The transmission properties of sapphire are generally good at the SCUBA-2 wavelengths, but are dependent on the thickness and ambient temperature. The total effective transmission values of the HWP integrated across the 850 and 450 µm filter bands are about 86% and 57% respectively [18].

The HWP rotates the incoming linear polarisation with twice the speed of the wave plate angle. The HWP is typically rotated at 2 Hz, providing a fast modulation of any linear polarisation by 8 Hz (see Equation 2.9). The data acquisition rate is ~175 Hz, yielding 20 samples per cycle. The atmosphere is stable on the order of 2 Hz and can be removed.

2.2 Instrumental Polarisation

At the angular resolution of JCMT, planets such as Uranus should appear as unpolarised point sources. In practice, however, POL-2 observations of such sources exhibit a measurable level of polarisation—albeit typically less than 1.5% at 850 μm. This is evidence that some part of the incoming astronomical radiation

Figure 2.5: The three blades that combine to form POL-2 are partially extended showing the two wire grids and the achromatic HWP. The two wire grids are the calibrator grid and the analyser grid. The rotating HWP is located between these two fixed grids. The calibrator grid is only inserted for test purposes. Stiffeners can be seen on all three blades. The one for the HWP is particularly thick. Their purpose is to reduce vibrations while the HWP spins.

is being partially polarised by one or more of the components of the telescope/POL-2/SCUBA-2 that are in the light path. This polarisation is referred to as "Instrumental Polarisation" (IP).

In order to establish the true Q and U from an astronomical source, it is necessary to correct for this effect. For the case of a low degree of polarisation in the incoming radiation and a low degree of IP, the following expressions provide good approximations to correct the measurement for the effects of the IP:

$$Q = Q_{\mathrm{m}} - I.ip_{\mathrm{q}} \tag{2.10}$$

$$U = U_{\mathrm{m}} - I.ip_{\mathrm{u}} \tag{2.11}$$

where Q_{m} and U_{m} are the measured values for a single bolometer sample at some point on the sky. Q and U are the true (corrected) values, I is the astronomical total intensity at the same point on the sky (i.e. the total intensity after removal of the sky and electronic backgrounds) and ip_{q} and ip_{u} are factors that may vary slowly with focal plane position and/or azimuth and elevation.

IP correction of a POL-2 map therefore requires the use of a total intensity map of the same area of the sky. This total intensity map is referred to as the IP reference map, and can now usually be derived from normal POL-2 observations as part of the standard data reduction process, obviating the need for an additional non-polarimetric reference map observation.

Whilst flat mirrors or surfaces will produce a small, constant polarisation across the beam, curved mirrors and other structures (for example the secondary mirror supports) will produce more complex polarisation effects, and these may distort the beam shape. Side-lobes can often show up with strong (typically 10–20%) polarisation but these effects are usually far from the main-beam. Calculations of typical antenna patterns for symmetrical Cassegrain antennas have not predicted strong polarisation in the main beam.

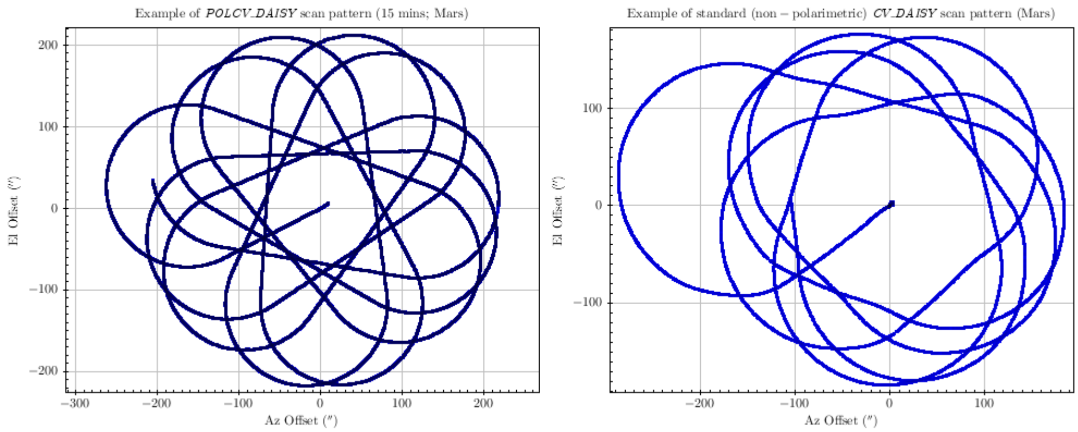


Figure 2.6: Left: Scan pattern from a typical SCUBA-2 CV_Daisy observation. Right: Scan pattern from a POL-2 Daisy. The standard POLCV_DAISY scan parameters are given in Table 2.1

The JCMT IP footprint is stronger than might be expected from the above considerations above (though typically less than 1.5% of the total intensity), and has the following distinctive features:

(1) the polarisation intensity is elevation dependent,

(2) there is ellipticity of the beam and it is elevation dependent, and

(3) the beam is elongated in the horizontal direction.

The dominant source of IP at the JCMT is the woven Goretex membrane, used as a wind blind. This membrane introduces both losses and polarisation. This effect is elevation dependent.

2.3 Observing mode

The standard POL-2 observing mode, POLCV_DAISY, is a "scan and spin" mode, in which the telescope is moving continuously in a Daisy-type pattern while the HWP spins.

The POLCV_DAISY scan mode is similar to the established Daisy scan mode routinely used for non-polarimetric SCUBA-2 observations of point-like or compact sources. However, it is slightly altered to allow for a slower telescope scanning speed.

The telescope must scan slowly enough to obtain sufficient data at each point on the sky to allow good Q and U values to be determined. The current commissioned scan pattern has a size of $200''$ and a scan speed of $8''$/s. The data reduction splits the data stream into short segments and determines a pair of Q and U values from each segment.

The length of each data segment is defined *primarily* by the time taken for the HWP to make a single rotation (0.5 s). The scan speed of $8''$/s was chosen to make this correspond to $4''$ on the sky. The default pixel size provided by `pol2map` is $4''$ at both 850 and 450 µm.

The standard POLCV_DAISY scan parameters are given in Table 2.1 and shown in Figure 2.7.

Parameter	Value
Half-wave plate-rotation frequency	2 Hz
Antenna scanning speed	8″/s
R_0 (map pattern radius)†	133″
R_t (turn radius)	99″
R_a (nominal avoidance radius)	77″

Table 2.1: The scan parameters used in the POLCV_DAISY mode. †This radius is *not* the size of the resulting map.

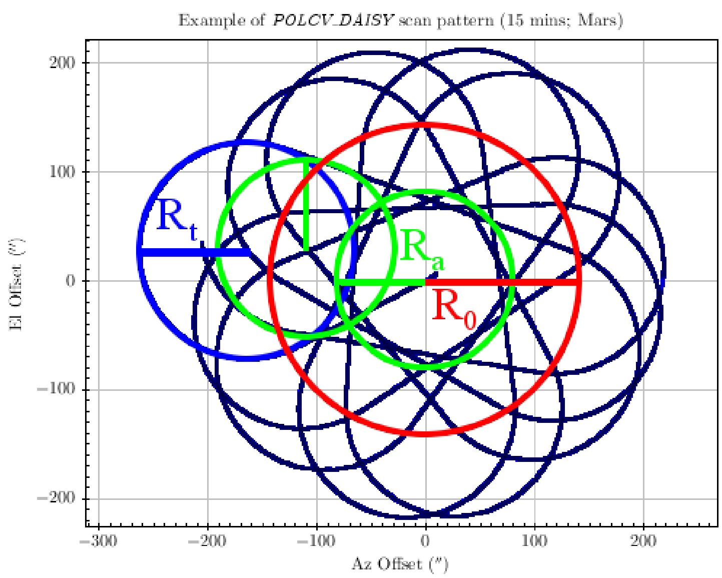


Figure 2.7: Detail of POLCV_DAISY. R_0 is the map pattern radius, R_t the turn radius, and R_a is the nominal avoidance radius. For more details see Table 2.1.

2.4 The raw data

SCUBA-2 is the detector instrument for POL-2, and as such, the raw data format of POL-2 data is the same as a typical SCUBA-2 observation. The sequence for both observations is:

(1) dark noise,
(2) flat-field,
(3) science scans, and
(4) flat-field.

The `SEQ_TYPE` keyword in the FITS header may be used to identify the nature of each scan. When a user searches for raw data from the CADC archive http://www3.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/jcmt/all of the data types listed above will be returned.

Critically, the `INBEAM` keyword in the FITS header may be used to identify if POL-2 is in the beam, and hence differentiate between SCUBA-2 and POL-2 observations.

Tip

Use the KAPPA command fitslist to see all FITS headers in a particular NDF. To obtain a specific header simply use the command fitsval:

```
% fitsval s8a20160112_00056_0001.sdf INBEAM
pol
```

The FITS header information may also be viewed via the GAIA `View / FITS header` drop-down menu option.

Shown below is an incomplete list of the raw files for a single sub-array (in this case s8a) for a short POL-2 observation. The first scan is the dark noise frame, and the second and last scans are the flat-field observations, which occur after the shutter opens to the sky at the start of the observation and closes at the end (note the identical file size); all of the scans in between are science scans.

```
% ls -lh /jcmtdata/raw/scuba2/s8a/20160112/00056


-rw-r--r-- 1 jcmtarch jcmt 5.6M Jan 12  2016 s8a20160112_00056_0001.sdf
-rw-r--r-- 1 jcmtarch jcmt 7.9M Jan 12  2016 s8a20160112_00056_0002.sdf
-rw-r--r-- 1 jcmtarch jcmt  25M Jan 12  2016 s8a20160112_00056_0003.sdf
-rw-r--r-- 1 jcmtarch jcmt  25M Jan 12  2016 s8a20160112_00056_0004.sdf
-rw-r--r-- 1 jcmtarch jcmt  25M Jan 12  2016 s8a20160112_00056_0005.sdf
...
-rw-r--r-- 1 jcmtarch jcmt  25M Jan 12  2016 s8a20160112_00056_0025.sdf
-rw-r--r-- 1 jcmtarch jcmt  25M Jan 12  2016 s8a20160112_00056_0026.sdf
-rw-r--r-- 1 jcmtarch jcmt  25M Jan 12  2016 s8a20160112_00056_0027.sdf
-rw-r--r-- 1 jcmtarch jcmt  22M Jan 12  2016 s8a20160112_00056_0028.sdf
-rw-r--r-- 1 jcmtarch jcmt 7.9M Jan 12  2016 s8a20160112_00056_0029.sdf
```

The SCUBA-2 data-acquisition (DA) system writes out a data file every 30 seconds; each of which contains 22 MB of data. The final file listed above is a flat-field scan, which will usually be smaller (7.9 MB in this example). Note that all of these files are written out eight times, once for each of the eight sub-arrays.

The main data array in each NDF is a cube, with the first two dimensions corresponding to bolometer columns and rows within a sub-array, and the third dimension corresponding to time slice index (sampled at roughly 200 Hz).

A standardised file naming scheme is used in which each file name starts with the sub-array name, followed by the UT date of the observation in the format `yyyymmdd`, followed by a five-digit observation number, followed by the sub-scan number. The name ends with the standard suffix `.sdf` used by all Starlink NDF data files. For instance, the files listed above hold data from the s8a sub-array for Observation 34 taken on 12$^{\text{th}}$ January 2016.

Units/Calibration

Raw POL-2 data come in uncalibrated units. The first calibration step is to scale the raw data to units of picowatts (pW) by applying the flat-field solution. This step is performed internally by the SMURF command calcqu—used to calculate I, Q, and U time-streams from the raw data—but can be done manually when examining the raw data.

If the purpose of a given POL-2 observation is to determine the percentage polarisations or vector angles within a source/region of interest, then the data may remain in pW. On the other hand, if the purpose is to establish the absolute polarised intensities, then a value for the Flux Conversion Factor (FCF) is required.

The resulting map may have the FCF applied to convert it into units of janskys. As is recommended with SCUBA-2 observing, it is advisable to check that the FCF value applied to the data is sensible (and must be done manually). For more details see Chapter 4.5.

Chapter 3
POL-2 Data Reduction – The Theory

3.1 The data flow

POL-2 data reduction is an involved process. A broad overview of this process is presented here first, followed by a discussion of the specific details. It should be noted that this same procedure is used irrespective of whether single or multiple observations are to be reduced.

The following description assumes that the makemap command is being used to create maps. Section 3.3 explains the effects of using the skyloop command in place of makemap.

The data reduction process can be broken down into three main stages – referred to as "Run 1", "Run 2" and "Run 3" in Figure 3.1.

Step 1

The initial step of the process (see Run 1 in Figure 3.1) creates a preliminary co-added total intensity (I) map from the raw data files for all observations provided to the reduction routine (see Chapter 4).

The process

The analysed intensity values in the raw data time-streams are first flat-fielded and converted into Q, U, and I time-streams using the calcqu command. These time-streams are stored for future use in the directory `qudata`, specified by the `QUDIR` parameter in the example command below.

The SMURF:makemap command is then used to create a separate map from the I time-stream for each observation, using SNR-based "auto-masking" to define the background regions that are to be set to zero at the end of each iteration. This step uses a PCA threshold of 50 (see Section 3.5 for more details).

```
pca.pcathresh = -50
```

These maps are stored for future use in the directory `maps`, specified by the `MAPDIR` parameter. Each map has a name of the form:

```
<UT_DATE>_<OBS_NUM>_<CHUNK_NUM>_imap.sdf
```

where <CHUNK_NUM> indicates the raw data file at the start of the contiguous chunk of data used to create the map, and is usually 0003.

A co-add is then formed by adding all these maps together. Each individual map is then compared with the co-add, in order to determine a pointing correction to be applied to the observation in future. These corrections are stored in the FITS header of the individual maps.

Step 2

In the second step of the process (see Run 2 in Figure 3.1) an improved I map is produced. These improvements come from

(1) applying the pointing corrections determined in Step 1,

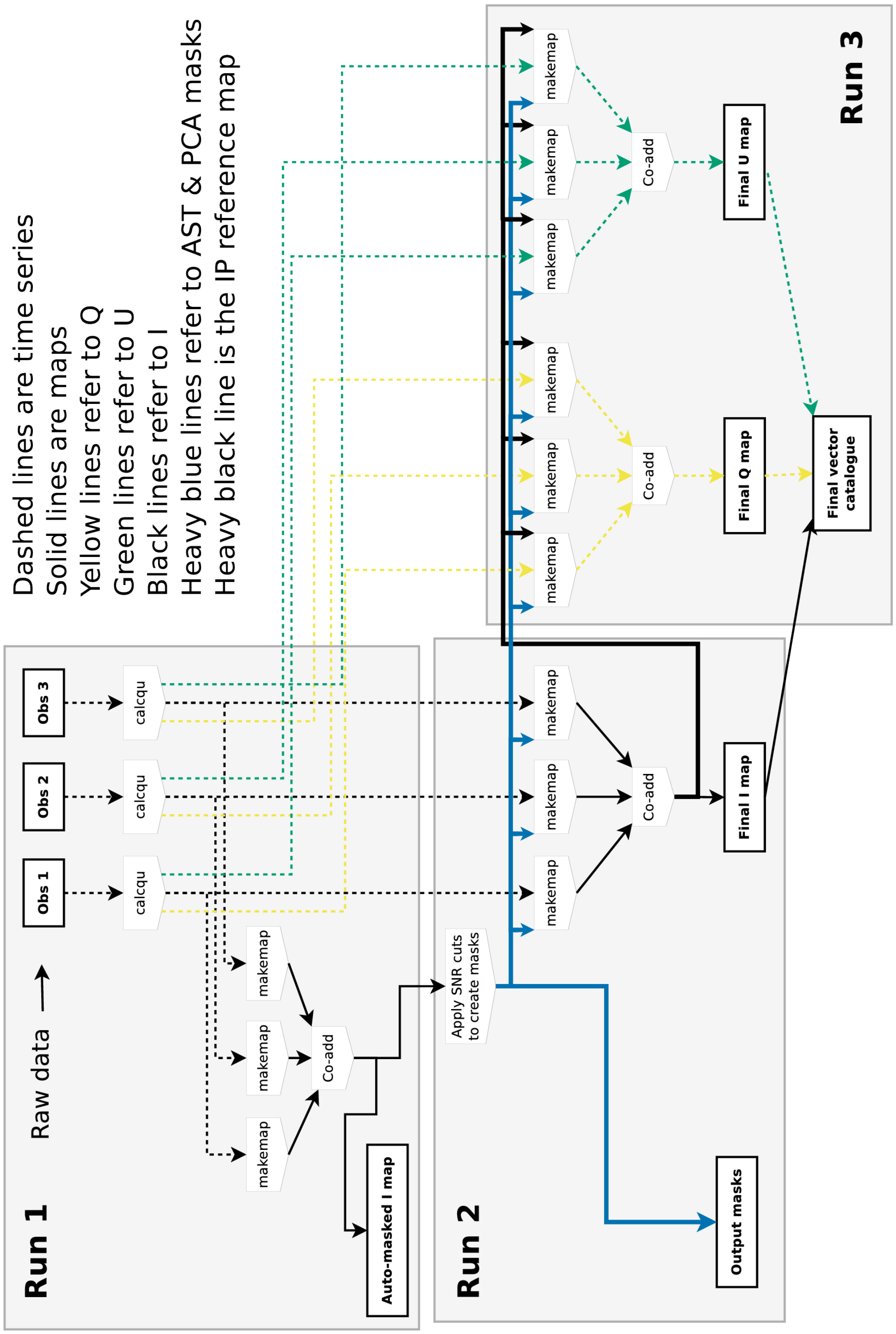


Figure 3.1: The data flow of the POL-2 data reduction method is presented. In this example, three POL-2 observations are reduced and combined in various stages and combination to produce I, Q, and U maps and a vector catalogue.

(2) the use of an increased number of PCA components (`pca.pcathresh=-150`), and

(3) using a single, fixed mask for all observations. The mask is determined from the preliminary co-added I map and thus includes fainter structure than would be used if the mask was based on only one observation.

Step 3

In the third step of the reduction process (see Run 3 in Figure 3.1), both the Q and U maps are produced. The production of the Q and U maps requires the Q and U time-series data (produced in Step 1), the final I map (produced in Step 2), and the output masks (also produced in Step 2). Once the Q and U maps are produced a final vector catalogue is created.

3.2 MAKEMAP

The POL-2 data reduction builds upon the existing SCUBA-2 Dynamic Iterative Map-Maker, hereafter just referred to as the "map-maker". This is the tool used to produce SCUBA-2 maps, and is invoked by the SMURF makemap command. It performs some pre-processing steps to clean the data, solves for multiple signal components using an iterative algorithm, and bins the resulting time-series data to produce a final science map.

In pol2map, the map-maker is used in conjunction with calcqu (see Section 3.4) to produce maps of Q and U, as well as I.

3.3 SKYLOOP

An option exists to use the SMURF skyloop command to generate maps in place of the SMURF makemap command. Each invocation of the makemap command creates a single map from the time-series data for a single observation. If multiple observations are processed together, multiple invocations of makemap are used and the resulting maps are co-added before moving on to the next step, as indicated in Figure 3.1. The skyloop command, on the other hand, will process multiple observations in a single invocation, creating a combined map from all observations. Thus, when using skyloop, Figure 3.1 would be effectively changed such that each occurrence of "three invocations of makemap feeding one co-add" would be replaced by a single invocation of "skyloop". See Section 3.7 for more information about the effects of using skyloop.

3.4 CALCQU

In addition to the POL-2 data reduction building on the existing SCUBA-2 map-maker, pol2map also relies on the SMURF command calcqu.

This calcqu tool creates time series holding I, Q, and U values from a set of POL-2 time series holding raw data values. The supplied time-series data files are first flat-fielded, cleaned and concatenated, before being used to create the I, Q, and U values. The I, Q, and U time-series are down-sampled to 2Hz (i.e. they contain two I, Q, or U samples per second), and are chosen to minimise the sum of the squared residuals between the measured raw data values and the expected values given by Equation 2.9.

3.5 PCA

One difference between the reduction of SCUBA-2 data and POL-2 data is the method used to remove the sky background. The sky background is usually very large compared with the astronomical signal, and both are subject to the same form of instrumental polarisation (IP—see Section 2.2). This IP acting on the high sky background values causes high background values in the Q and U maps. However, there is evidence that the IP is not constant across the focal plane, resulting in spatial variations in the background of the Q and U maps.

For non-POL-2 data, the background is removed using a simple common-mode model, in which the mean of the bolometer values is found at each time slice and is then removed from the individual bolometer values. This ignores any spatial variations in the background, and so fails to remove the background properly in POL-2 Q and U maps.

To fix this, a second stage of background removal is used when processing POL-2 data, following the initial common-mode removal. This second stage is based upon a Principal Component Analysis (PCA) of the 1280 time-streams in each sub-array (the Q and U data are processed separately). The PCA process identifies the strongest time-dependent components that are present within multiple bolometers. These components are assumed to represent the spatially varying background signal and are removed, leaving just the astronomical signal. You may specify the number of components to remove, via a makemap configuration parameter called pca.pcathresh although pol2map, the reduction command for POL-2 data, provides suitable defaults for this parameter.

- first stage uses `pca.pcathresh` = -50
- second stage uses `pca.pcathresh` = -150

On each makemap iteration, the PCA process removes the background (thus reducing the noise in the map) but also removes some of the astronomical signal. The amount of astronomical signal removed will be greater for larger values of `pca.pcathresh`. However, this astronomical signal is still present in the original time-series data, and so can be recovered if sufficient makemap iterations are performed. In other words, the use of larger values of `pca.pcathresh` slows down the rate at which astronomical signal is transferred from the time-series data to the map, thus increasing the number of iterations required to recover the full astronomical signal in the map.

Spatial variations in the sky background may also be present in non-POL-2 data, but at a lower level. For a discussion of why PCA is not routinely run on non-polarimetric SCUBA-2 data, see Appendix A.

3.6 Masking

A mask is a two-dimensional array that has the same shape and size as the final map, and that is used to indicate where the source is expected to fall within the map. 'Bad' pixel values within a mask indicate background pixels, and 'good' pixel values indicate source pixels. Masks are used for two main purposes:

(1) They prevent the growth of gradients and other artificial large scale structures within the map. For this purpose, the astronomical signal at all background pixels defined by the mask is forced to zero at the end of each iteration within makemap (except for the final iteration).

(2) They prevent bright sources polluting the evaluation of the various noise models (PCA, COM, FLT) used within makemap. Source pixels are excluded from the calculation of these models.

The pol2map script uses different masks for these two purposes—the "AST" mask and the "PCA" mask. The PCA mask is, in general, less extensive than the AST mask, with the source areas being restricted to the brighter inner regions. Each of these two masks can either be generated automatically within pol2map, or be specified by a fixed external NDF.

3.7 Tailoring a reduction

Variances between POL-2 maps

MAPVAR is a pol2map parameter that controls how the variances in the co-added I, Q, and U maps are formed.

If MAPVAR is set TRUE, the variances in the co-added I, Q, and U maps are formed from the spread of pixel data values between the individual observation maps. If MAPVAR is FALSE (the default), the variances in the co-added maps are formed by propagating the pixel variance values created by makemap from the individual observation maps (these are based on the spread of I, Q, or U values that fall within each pixel).

Use MAPVAR=TRUE only if enough observations are available to make the variances between them meaningful. A general lower limit on its value is difficult to define, but a minimum of 10 observations is advised.

If a test of the effect of this option is required on a field for which the I, Q, and U maps from a set of individual observations are already available, the following may be done:

```
% pol2map in=maps/\* iout=imapvar qout=qmapvar uout=umapvar mapvar=yes \
                   ipcor=no cat=cat_mapvar debias=yes
```

assuming that the I, Q, and U maps are in directory maps. The variances in imapvar.sdf, qmapvar.sdf and umapvar.sdf will be calculated using the new method, and these variances will then be used to form the errors in the cat_mapvar.FIT catalogue.

In general, within the source regions, the variances created using MAPVAR=TRUE will be larger than those created using using MAPVAR=FALSE (within background regions there should be little difference). This is partly caused by residual uncorrected pointing errors, which have a particularly large effect near bright point sources if MAPVAR=TRUE.

It is also partly caused by intrinisic instabilities within the iterative map-making algorithm, which allow low-level artificial extended structures to develop within the source regions defined by the AST mask. Such artificial structures will vary from observation to observation, and so will contribute to the variances calculated using MAPVAR=TRUE.

Two options are provided by pol2map that may be useful in reducing the larger-than-expected dispersion between maps made from different observations.

(1) Setting the parameter OBSWEIGHT=TRUE when running pol2map will cause each observation to be assigned a separate weight, which will be used when forming the co-add of all observations. This will affect both the data values and the variances in the resulting co-add. The purpose of these weights is to down-weight observations that produce maps that are very dissimilar to maps made from the other observations.

Without this parameter setting, the co-add is formed using weights equal to the reciprocal of the pixel variance values in each individual observation's map. As mentioned above, these pixel variance values can sometimes seriously underestimate the dispersion between observations. For instance, observations that are clearly bad (e.g. out of focus) can have relatively low pixel variance values, and thus be included with high weights in the final co-add.

If the `OBSWEIGHT` parameter is set `TRUE`, each observation is given an additional weight that is used to factor the per-pixel weights derived from the pixel variance values, in order to down-weight observations that are clearly bad. To form these weights, an initial co-add is formed using equal weights for all observations. The maps made from the individual observations are then compared with this initial co-add, and each observation is assigned a weight equal to the reciprocal of the mean squared residual between the individual observation's map and the initial co-add (any required pointing correction is applied to the individual observation map before forming these residuals). The calculation of the mean squared residual is limited to those pixels inside the AST mask (i.e. source pixels). The weights derived in this manner are normalised to have a median value of 1.0, and any normalised weights larger than 1.0 are reduced to 1.0. An improved co-add is then formed using these observation weights.

Another iteration is then performed, in which individual maps are compared with this improved co-add and new weights are derived. This iterative process continues until the typical error in the middle of the co-add stops falling significantly.

(2) Setting the parameter `SKYLOOP=TRUE` when running pol2map will cause maps to be made using the SMURF:skyloop command, instead of makemap. In the context of the skyloop documentation, one "chunk" of data usually corresponds to a single observation.

A single invocation of skyloop creates an I, Q, or U map from all supplied observations, using a method that attempts to minimise the intrinsic instabilities of the map-making algorithm within the AST mask. It should be noted that convergence can require a significantly greater number of iterations when using skyloop than when using makemap. Also, skyloop requires much more disk space than makemap.

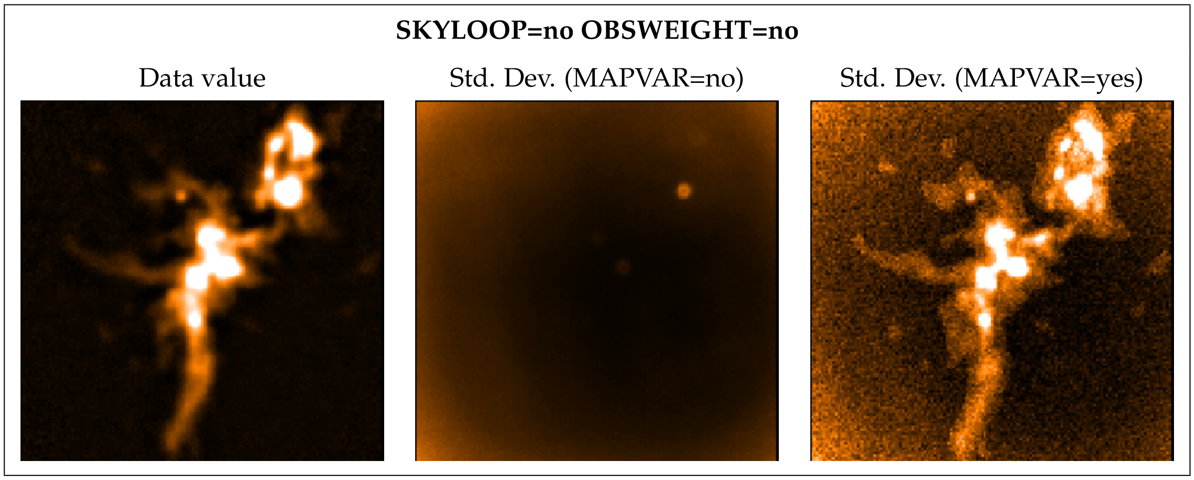
The skyloop command combines all observations together at each iteration of the map-making algorithm. Since the spurious large-scale structures created at each iteration are independent of each other, taking the mean of the maps after each iteration reduces the level of such structures, and prevents them from growing in amplitude on successive iterations due to the instability in the map-making algorithm.

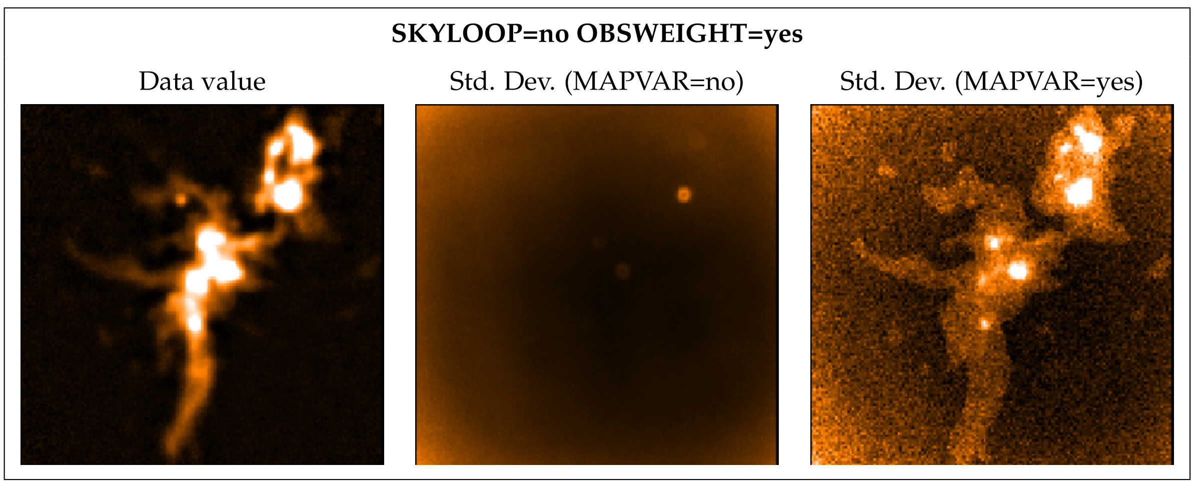
The above two methods can be used together by supplying `TRUE` values for both `OBSWEIGHT` and `SKYLOOP`.

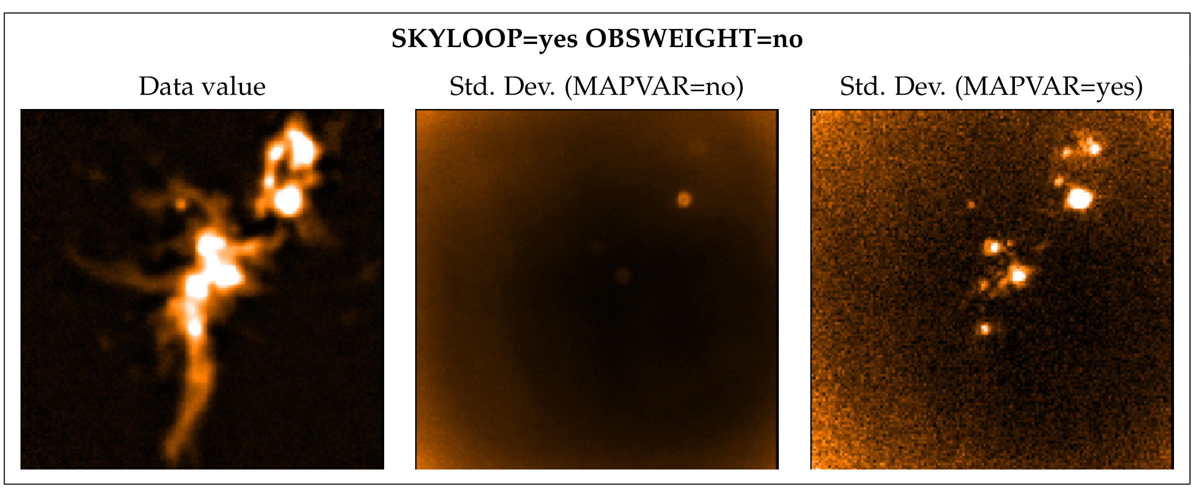
Note, it is not recommended to use `MAPVAR=TRUE` or `SKYLOOP=TRUE` on Step 1 (i.e. when creating the auto-masked maps). Doing so is of no benefit to the final maps and can cause problems such as negative bowling.

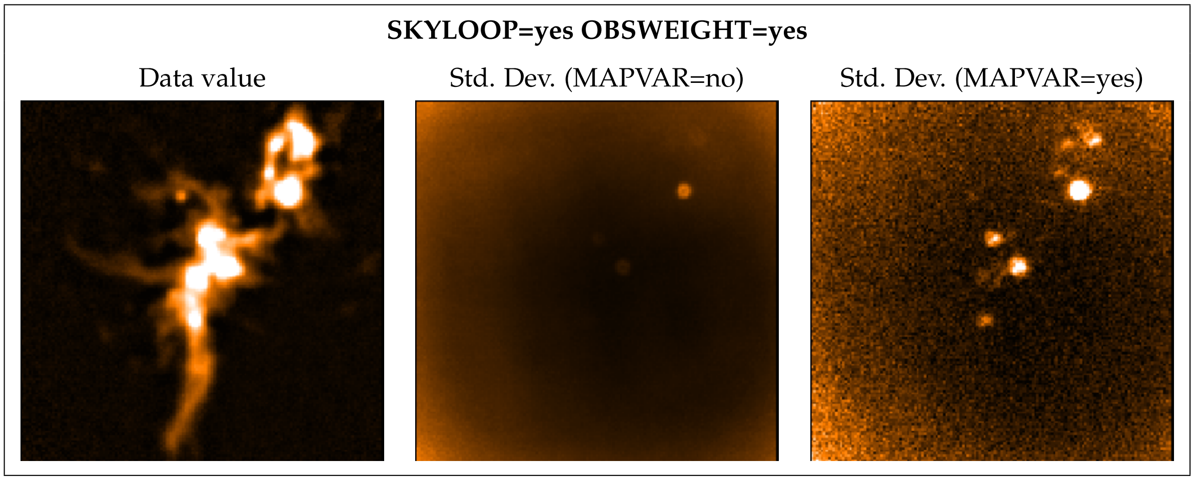
If the `OBSWEIGHT` parameter is used at Steps 2 or 3, then it must also be used at Step 1.

The following panels show the effects of using `SKYLOOP` and `OBSWEIGHT` on a total-intensity mosaic of 21 observations. All data-value maps are shown with a single scaling, and all standard-deviation maps are shown with a single scaling (different from the scaling for the data-value maps).

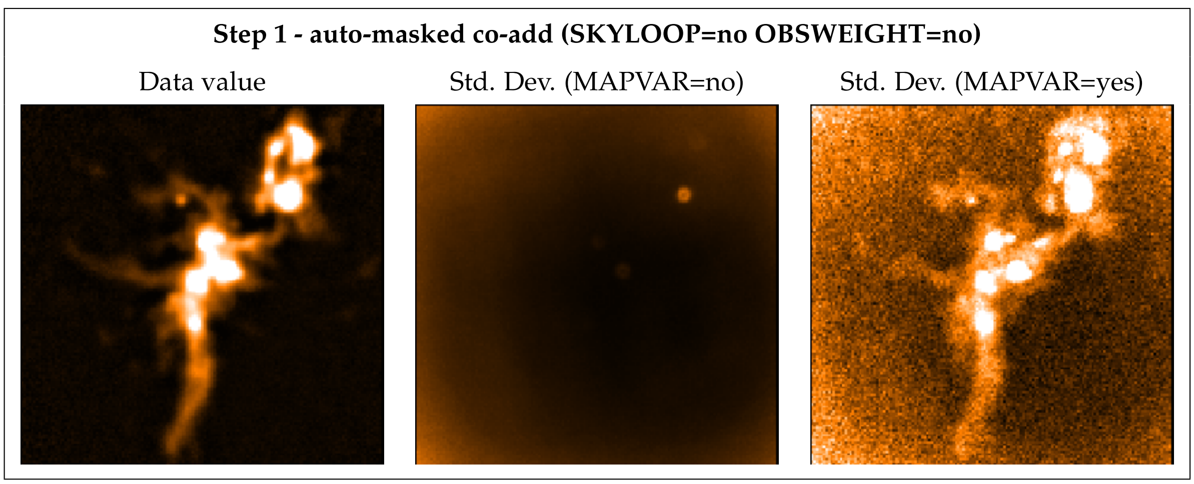

SKYLOOP=no OBSWEIGHT=no
Data value
Std. Dev. (MAPVAR=no)
Std. Dev. (MAPVAR=yes)


SKYLOOP=no OBSWEIGHT=yes
Data value
Std. Dev. (MAPVAR=no)
Std. Dev. (MAPVAR=yes)


SKYLOOP=yes OBSWEIGHT=no
Data value
Std. Dev. (MAPVAR=no)
Std. Dev. (MAPVAR=yes)



For comparison, below are the equivalent auto-masked maps made by Step 1.



Chapter 4
POL-2 Data Reduction – Running pol2map

The previous chapter, Chapter 3, described how pol2map produces I, Q, and U maps from raw POL-2 data. It showed that this reduction process—-which uses pol2map—comprises three steps.

As with the other Python scripts in SMURF, it is possible to get more information about the available parameters by running either:

```
% pol2map --help
```

or the

```
% smurfhelp pol2map
```

command.

4.1 How to use pol2map

Before running pol2map directly, it is necessary to ensure that the Starlink environment has been initialised and the SMURF package started (see Section 1.3.2 and Section 1.3.3).

This chapter describes how to run pol2map firstly to produce an initial I map; and then again to produce the final I, Q, and U maps and a vector catalogue, as described in Section 4.2.

To run pol2map, values should normally be supplied for the following command-line parameters[1], in order to produce the initial intensity image. Note that if a parameter description ends with a value in square brackets, it is the default value that will be used for the parameter if no value is supplied on the command line.

- `IN` A list of input NDFs containing raw POL-2 data. There are many ways in which the list of files can be supplied, as described in the "Specifying Groups of Objects" section of SUN/95. The easiest of these is to create a simple text file containing the names of the raw data files (one per line), and then supply the name of the text file, preceded by an up-caret character (`^`), as the value for parameter `IN`. Note that the specified names of the raw data files can contain wildcards such as "∗" and "?".
- `IOUT` The name of the NDF in which to store the total intensity (I) map (in pW) incorporating all supplied observations. The supplied file name should either have a file type of `.sdf`, or no file type at all (in which case `.sdf` will be appended to the supplied value). Any existing file with the same name will be overwritten.
- `QOUT` The output NDF in which to return the Q map incorporating all supplied observations. This will be in units of pW. Null (`!`) should be supplied if no Q map is required.

[1] Note the distinction between "command-line parameters" that are supplied on the pol2map command line, and "configuration parameters" that are specified within a configuration file. Values for all *configuration* parameters are obtained using a single *command-line* parameter called `CONFIG`.

UOUT The output NDF in which to return the U map incorporating all supplied observations. This will be in units of pW. Null (!) should be supplied if no U map is required.

MAPDIR The name of the directory in which to place the Q, U, and I maps made from each individual observation supplied via IN, before co-adding them. If null (!) is supplied, the new maps are placed in the same temporary directory (chosen automatically) as all the other intermediate files, and so will be deleted when the script exits (unless the parameter RETAIN is set to be TRUE). Note that these maps are always in units of pW. Each one will contain FITS headers specifying the pointing corrections needed to align the map with the reference map. [!]

QUDIR The name of a directory in which to place the Q, U, and I time series generated by SMURF calcqu, prior to generating maps from them. If null (!) is supplied, they are placed in the same temporary directory as all the other intermediate files, and so will be deleted when the script exits (unless the parameter RETAIN is set to be TRUE). [!]

Some additional command-line parameters are required when pol2map is used for the second time—as discussed in Section 4.3—to produce the final I, Q, U maps and vector catalogue[2].

CAT The output FITS vector catalogue. No catalogue is created if null (!) is supplied. Note that, by default, the Q, U, and I_p values in this catalogue will be in units of mJy/beam. [!]

MASK Specifies the type of masking to be used within makemap (the same type of masking is used to create all three maps—I, Q, and U).

MASKOUT1 If a non-null value is supplied for MASKOUT1, it specifies the NDF in which to store the AST mask created from the NDF specified by Parameter MASK. Only used if an NDF is supplied for Parameter MASK. [!]

MASKOUT2 If a non-null value is supplied for MASKOUT2, it specifies the NDF in which to store the PCA mask created from the NDF specified by Parameter MASK. Only used if an NDF is supplied for Parameter MASK. [!]

IPREF The total intensity map to be used for IP correction. The map must be in units of pW. If the same value is supplied for both IOUT and IPREF, the output I map will be used for IP correction. [!]

DEBIAS TRUE if a correction for statistical bias is to be made to percentage polarisation and polarised intensity in the output vector catalogue specified by the parameter CAT. [FALSE]

The pol2map command provides many other parameters that may be used to modify its behaviour in various ways. To see a full list, type the following.

```
% pol2map --help
```

4.2 pol2map – producing the initial I map

As discussed in Chapter 3, pol2map must first be run on the raw data to produce an initial I map. In this first step:

[2] This second usage of pol2map includes both "Run 2" and "Run 3" in Figure 3.1.

```
% pol2map in=^myfiles.list iout=iauto qout=! uout=! mapdir=maps qudir=qudata
```

where the file `myfiles.lis` contains a list of the raw data files to be included in the map, and could (for instance) look like this.

```
% cat myfiles.lis
/jcmtdata/raw/scuba2/s8a/20160125/00043/*
/jcmtdata/raw/scuba2/s8b/20160125/00043/*
/jcmtdata/raw/scuba2/s8c/20160125/00043/*
/jcmtdata/raw/scuba2/s8d/20160125/00043/*
```

This uses all available data from all four 850 µm sub-arrays, for Observation 43 taken on 2016 January 25[3]. In addition, the data used in this example also come from Observations 56 and 59 taken on 2016 January 11 (UT).

Tip

An up-caret (ˆ) is required any time the user is reading in a group text file in Starlink. For the map-maker, this includes the configuration file (a group of configuration parameters) and the list of input files (a group of NDFs *e.g.* `in=ˆ myfiles.lis`).

Note that the pol2map script also invokes the SMURF pol2check command, in order to first ensure that the input files are actually POL-2 files.

Note that `qout` and `uout` are set to null values, as no Q or U maps are required to be produced during this initial Step 1 reduction stage.

The following shows the output from running this initial pol2map command.

```
Logging to file pol2map.log
Calculating Q, U and I time streams from raw analysed intensity data...
   1/3: Processing 116 raw data files from observation 20160125_00043 ...
   2/3: Processing 116 raw data files from observation 20160112_00059 ...
   3/3: Processing 116 raw data files from observation 20160112_00056 ...

>>>>   Making I map from 20160125_00043_0003...

>>>>   Making I map from 20160112_00056_0003...

>>>>   Making I map from 20160112_00059_0003...

Co-adding I maps from all observations:

20160125_00043_0003: Storing pointing corrections of (0.0,0.0) arc-seconds
for future use

20160112_00056_0003: Storing pointing corrections of (1.9,2.8) arc-seconds
for future use

20160112_00059_0003: Storing pointing corrections of (2.1,2.4) arc-seconds
for future use
```

[3] The input files should all be for a single waveband from one or more POL-2 observations. Users should not mix files from different wavebands and/or astronomical regions.

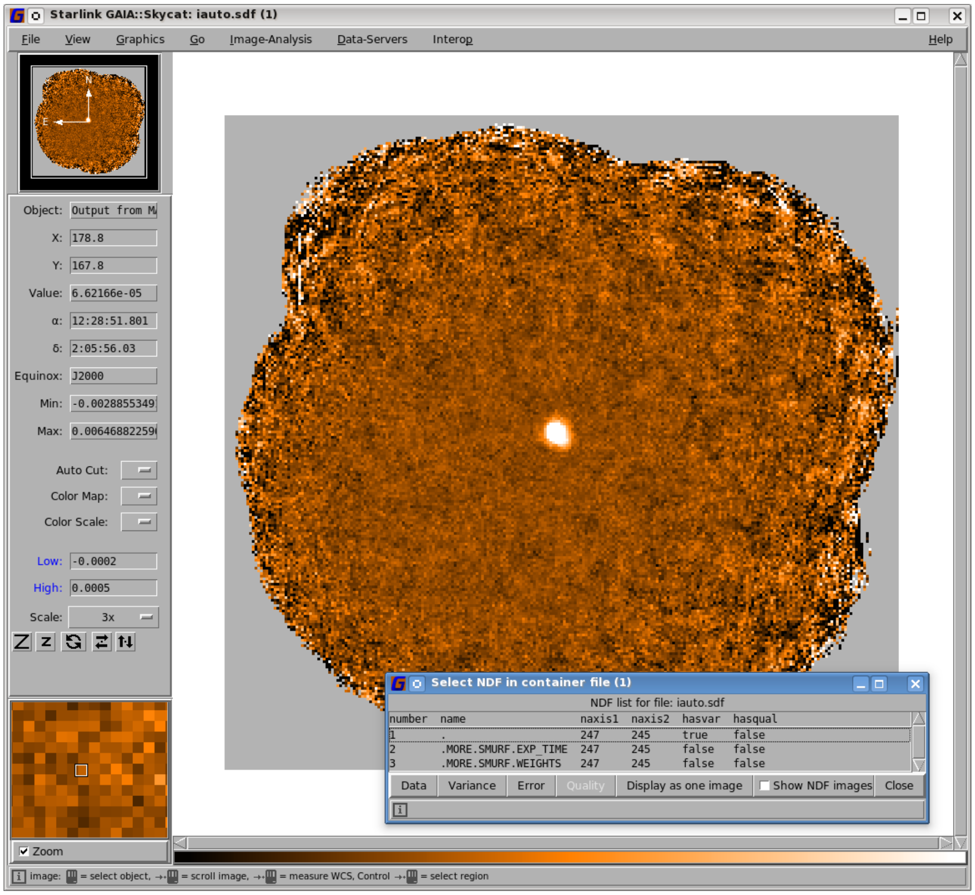


Figure 4.1: The I map, `iauto.sdf`, as viewed with GAIA.

The files and folders produced by this reduction process are described below.

`pol2map.log` A log file containing the output from the various SMURF, KAPPA and POLPACK commands run as part of the pol2map command (pol2map is actually a Python script that runs various other Starlink tasks behind the scenes in order to perform the bulk of the work).

`qudata/` A folder containing the I, Q, and U time-series data for each sub array for each observation. These are produced by calcqu (see Section 3.4).

`maps/` A folder containing the individual I maps from each separate observation. These will have names that end with `_imap.sdf`.

`iauto.sdf` Output total intensity map. The included term "auto" is used to indicate that it was created using an automatically generated AST mask.

The output I map, `iauto.sdf`, can be opened and viewed with GAIA.

The `maps` folder contains the individual I maps from each separate observation:

`20160112_00056_0003_imap.sdf` `20160112_00059_0003_imap.sdf` `20160125_00043_0003_imap.sdf`

and the `qudata` folder contains these files.

```
s8a20160112_00056_0003_IT.sdf  s8b20160112_00059_0003_IT.sdf  s8c20160125_00043_0003_IT.sdf
s8a20160112_00056_0003_QT.sdf  s8b20160112_00059_0003_QT.sdf  s8c20160125_00043_0003_QT.sdf
s8a20160112_00056_0003_UT.sdf  s8b20160112_00059_0003_UT.sdf  s8c20160125_00043_0003_UT.sdf
s8a20160112_00059_0003_IT.sdf  s8b20160125_00043_0003_IT.sdf  s8d20160112_00056_0003_IT.sdf
s8a20160112_00059_0003_QT.sdf  s8b20160125_00043_0003_QT.sdf  s8d20160112_00056_0003_QT.sdf
s8a20160112_00059_0003_UT.sdf  s8b20160125_00043_0003_UT.sdf  s8d20160112_00056_0003_UT.sdf
s8a20160125_00043_0003_IT.sdf  s8c20160112_00056_0003_IT.sdf  s8d20160112_00059_0003_IT.sdf
s8a20160125_00043_0003_QT.sdf  s8c20160112_00056_0003_QT.sdf  s8d20160112_00059_0003_QT.sdf
s8a20160125_00043_0003_UT.sdf  s8c20160112_00056_0003_UT.sdf  s8d20160112_00059_0003_UT.sdf
s8b20160112_00056_0003_IT.sdf  s8c20160112_00059_0003_IT.sdf  s8d20160125_00043_0003_IT.sdf
s8b20160112_00056_0003_QT.sdf  s8c20160112_00059_0003_QT.sdf  s8d20160125_00043_0003_QT.sdf
s8b20160112_00056_0003_UT.sdf  s8c20160112_00059_0003_UT.sdf  s8d20160125_00043_0003_UT.sdf
```

4.3 pol2map – producing the I, V, U maps and catalogue

As discussed in Chapter 3, the I map output from the initial run of pol2map is used to derive the final I, Q, and U maps. If requested, a vector catalogue is also produced.

The second and third steps of the POL-2 data reduction process can be run via a single command.

```
% pol2map in=qudata/\* iout=iext qout=qext uout=uext mapdir=maps mask=iauto \
          maskout1=astmask maskout2=pcamask ipref=iext cat=mycat debias=yes
```

The following shows the output from running this second pol2map command. First, pol2map produces new I maps for each map, correcting the position using the correction stored in the old I map[4], and then co-adds all the observations.

```
Logging to file pol2map.log
(existing file pol2map.log moved to pol2map.log.1)

Masking will be based on SNR values in 'iauto'.

>>>>   Making I map from 20160112_00056_0003...

   Using pre-calculated pointing corrections of (1.9,2.8) arc-seconds

>>>>   Making I map from 20160125_00043_0003...

   Using pre-calculated pointing corrections of (0.0,0.0) arc-seconds

>>>>   Making I map from 20160112_00059_0003...

   Using pre-calculated pointing corrections of (2.1,2.4) arc-seconds
Coadding I maps from all observations:
```

As pol2map continues, the Q and U maps are produced, again with pointing corrections. This is followed by the creation of the output vector catalogue.

```
>>>>   Making Q map from 20160112_00056_0003...

   Using pre-calculated pointing corrections of (1.9,2.8) arc-seconds
```

[4] This correction is found by aligning the old I map with the `iauto.sdf` map.

```
>>>>   Making Q map from 20160125_00043_0003...

   Using pre-calculated pointing corrections of (0.0,0.0) arc-seconds

>>>>   Making Q map from 20160112_00059_0003...

   Using pre-calculated pointing corrections of (2.1,2.4) arc-seconds
Coadding Q maps from all observations:

>>>>   Making U map from 20160112_00056_0003...

   Using pre-calculated pointing corrections of (1.9,2.8) arc-seconds

>>>>   Making U map from 20160125_00043_0003...

   Using pre-calculated pointing corrections of (0.0,0.0) arc-seconds

>>>>   Making U map from 20160112_00059_0003...

   Using pre-calculated pointing corrections of (2.1,2.4) arc-seconds
Coadding U maps from all observations:
Creating the output catalogue: 'mycat'...

45604 vectors written to the output catalogue.
```

The output of this final run of pol2map is as follows.

`pol2map.log` A log file containing the output from the pol2map command. Note previous log files are moved to a new name such as `pol2map.log.1`.

`astmask.sdf` The AST mask used in the creation of the final *I*, *Q*, and *U* maps.

`pcamask.sdf` The PCA mask used in the creation of the final *I*, *Q*, and *U* maps.

`iext.sdf` The total intensity image, created using the external AST and PCA masks described above.

`qext.sdf` The *Q* map (*i.e* the intensity of the radiation linearly polarised in the direction parallel or perpendicular to the reference plane), created using an external AST and PCA mask.

`maps/` A folder containing the individual *I*, *Q*, and *U* maps from each separate observation. These will have names that end with `_Imap.sdf`, `_Qmap.sdf`, or `_Umap.sdf`.

`uext.sdf` The *U* map (*i.e.* the intensity of the radiation linearly polarised in the direction $\pm 45^\circ$ to the reference plane).

`mycat.FIT` The output vector catalogue, containing a range of values derived by pol2map for each pixel contained within the *I* map.

The maps folder now contains individual *Q* and *U* maps, alongside the existing *I* maps listed below.

```
20160112_00056_0003_Imap.sdf  20160112_00059_0003_Imap.sdf  20160125_00043_0003_Imap.sdf
20160112_00056_0003_Qmap.sdf  20160112_00059_0003_Qmap.sdf  20160125_00043_0003_Qmap.sdf
20160112_00056_0003_Umap.sdf  20160112_00059_0003_Umap.sdf  20160125_00043_0003_Umap.sdf
20160112_00056_0003_imap.sdf  20160112_00059_0003_imap.sdf  20160125_00043_0003_imap.sdf
```

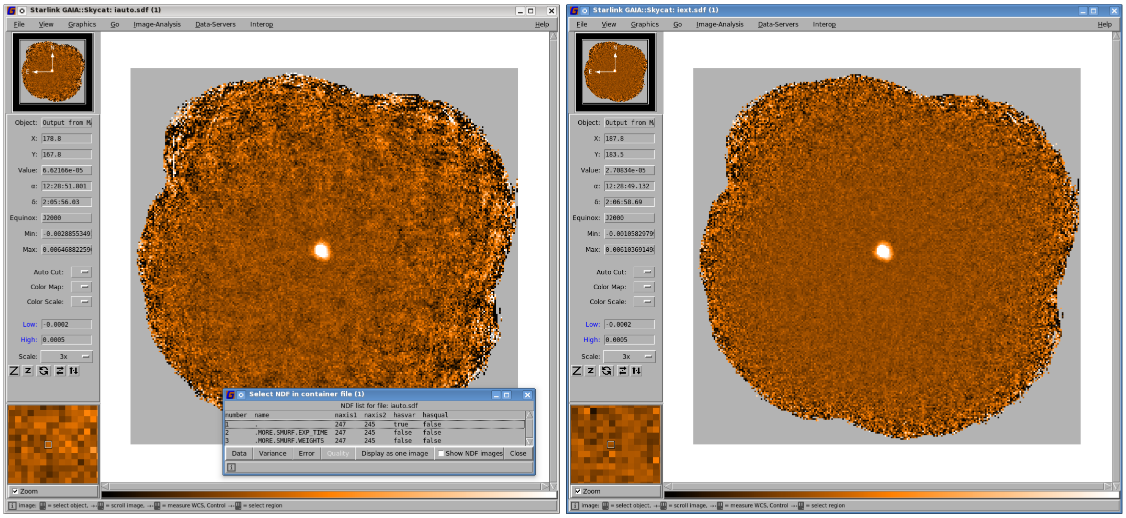


Figure 4.2: Left: I map, `iauto`, as produced by the automask on the first pass of pol2map. Right: Final I map, `iext`, as viewed with GAIA. The flatter background is due to the increase in `pca.pcathresh`.

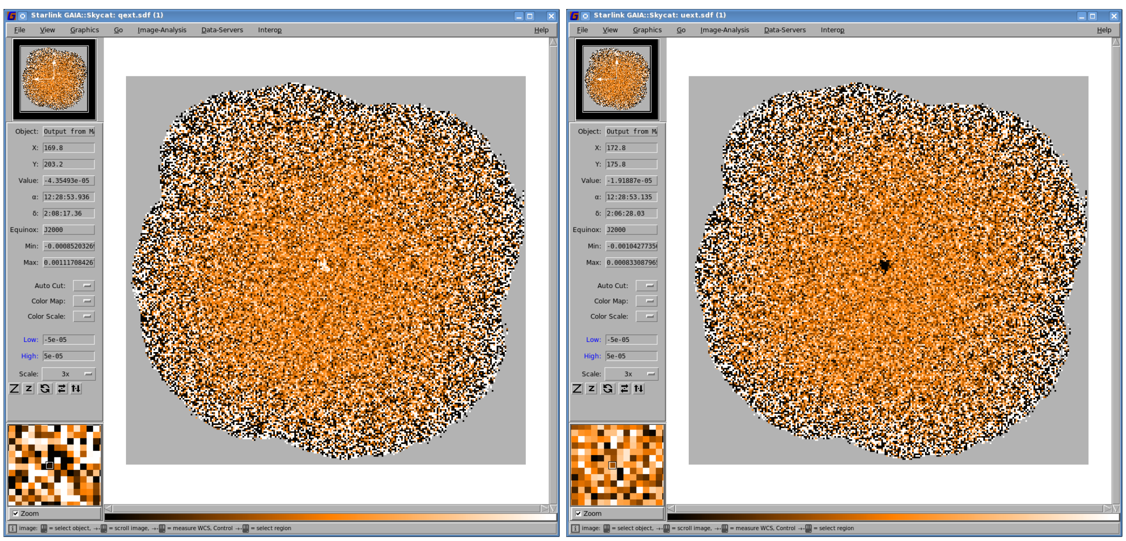


Figure 4.3: Left: Q map, `qext.sdf`. Right: U map `uext.sdf`, as viewed with GAIA.

4.4 Output vectors from pol2map

The output vector catalogue contains a range of values derived by pol2map for each pixel contained within the I map. Intensity values and errors in the catalogue are expressed in units of mJy/beam. If desired, it is possible to switch the catalogue to units of pW by using `Jy=no` on the pol2map command line. The columns are listed below.

X Pixel coordinate at the centre of the pixel

Y Pixel coordinate at the centre of the pixel

RA RA coordinate at the centre of the pixel

Dec Dec coordinate at the centre of the pixel

I Total intensity

DI Error in **I**

Q Stokes Q parameter

DQ Error in **Q**

U Stokes U parameter

DU Error in **U**

P Percentage polarisation

DP Error in **P**

ANG Angle of polarisation

DANG Error in **ANG**

PI Polarised intensity (I_p)

DPI Error in polarised intensity

AST Integer values that indicate if the corresponding vector position is inside the AST mask. If the vector is outside the mask, the corresponding column for the mask will have a blank/null value. Vectors that are inside a mask will have a non-zero integer value for the corresponding column. Each "island" within a mask (i.e. a contiguous group of source pixels) will have a different integer value assigned, starting at 1. All vectors within an island are assigned the integer value of the island.

PCA Contains integer values that indicate if the corresponding vector position is inside the PCA mask. Value conventions are the same as for the **AST** column (above).

4.5 POL-2 FCFs

Inserting POL-2 in front of SCUBA-2 reduces the throughput to SCUBA-2. POL-2 is not a perfect polarimeter. Its wire grid absorbs and scatters incoming signal so the modulation amplitude is lower than for a perfect polarimeter. In addition cross polarisation and depolarisation decreases the modulation amplitude without decreasing the power in the transmitted signal. The first type of inefficiencies can be measured by comparing normal SCUBA-2 maps with and without the polarimeter inserted. Such observations have been done on Uranus, Mars and Jupiter. The second type of losses can be measured with a source of known polarisation.

To convert POL-2 data to astronomical units such as mJy/beam a Flux Conversion Factor, FCF, must be applied to the data. For POL-2, the FCFs are quoted in terms of the non-polarimetric SCUBA-2 FCFs.

POL-2 FCFs are now applied automatically within pol2map when catalogue columns in mJy/beam are required by the user, i.e. if the pol2map parameter `JY` is set to `TRUE` (the default option). The current version of pol2map uses the new, revised POL-2 FCFs for 850 µm and 450 µm and takes account of cases where the supplied raw data span one of the dates at which the FCF changed (20180630 and 20161101).

4.6 Changing pixel size in pol2map

Inevitably, as with unpolarised SCUBA-2 data reduction, users will sometimes find it necessary to fine-tune the pol2map reduction process for specific situations.

The bin size within the final vector catalogue is controlled by the BINSIZE parameter in the SMURF pol2map command.

```
% pol2map binsize=12
```

Changing the catalogue bin size in this way does not change the pixel size of the maps created pol2map. Instead, the maps are binned up to the requested bin size before the catalogue is created. There is another parameter, called PIXSIZE, which controls the map pixel size, but it is usually advisable to leave this at its default value. This is because the map pixel size can affect the behaviour of the iterative algorithm used to create maps.

Chapter 5
POL-2 Image Display

5.1 GAIA

The Starlink package GAIA can be used to inspect the results of the data reduction. To plot the output vector catalogue onto the final total intensity map, first open up the I map in GAIA.

```
% gaia iext.sdf
```

From the main GAIA window, select the drop-down menu option **Image Analysis / Polarimetry toolbox**. This should launch a new toolbox window entitled **GAIA: Polarimetry**. From this window, use the drop-down menu option **File / Open** to load the file `mycat.FIT`. This should then populate the lower part of the window with the contents of this polarimetry catalogue file. Each of the vectors in this file will be automatically overlaid on the main image window (see Figure 5.1).

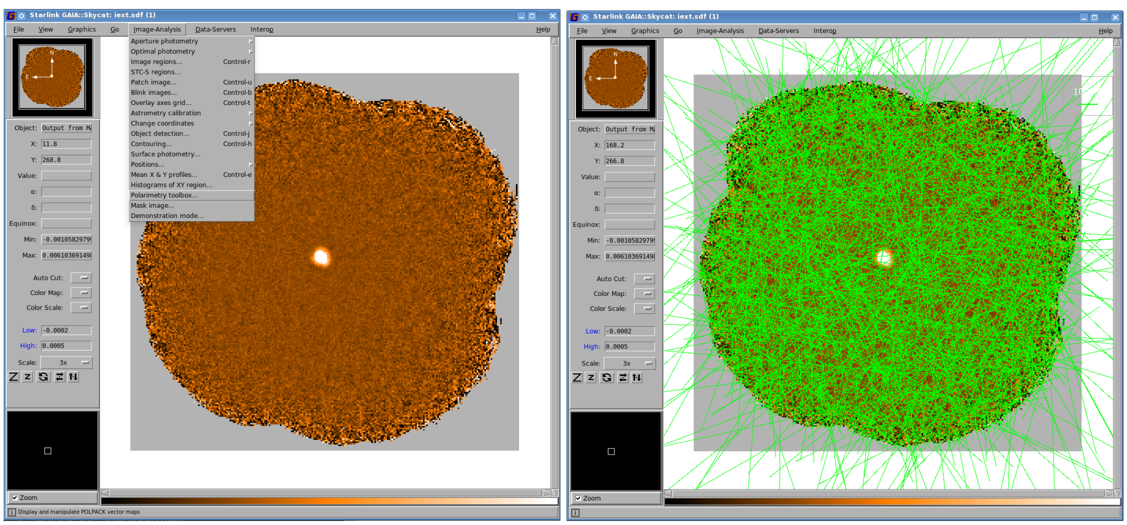


Figure 5.1: Left: Opening up the polarimetry toolbox in GAIA. Right: The initial POL-2 vectors overplotted in GAIA.

In order to filter the number of overlaid vectors down to a more useful number and size, the various options in the **GAIA: Polarimetry** toolbox can be used. First, select the **Rendering** tab on the left hand side. This will reveal a panel that will indicate which quantities are currently being used for the vector overlays. In this case, the vector length is taken from the **P** column of the table, and the vector angles are taken from the **ANG** column.

Currently the figure has too many vectors to be scientifically meaningful. To filter out most of the extraneous vectors, click on the **Selecting** tab, and set the **Expression** field to be the following:

```
$I/$DI<10
```

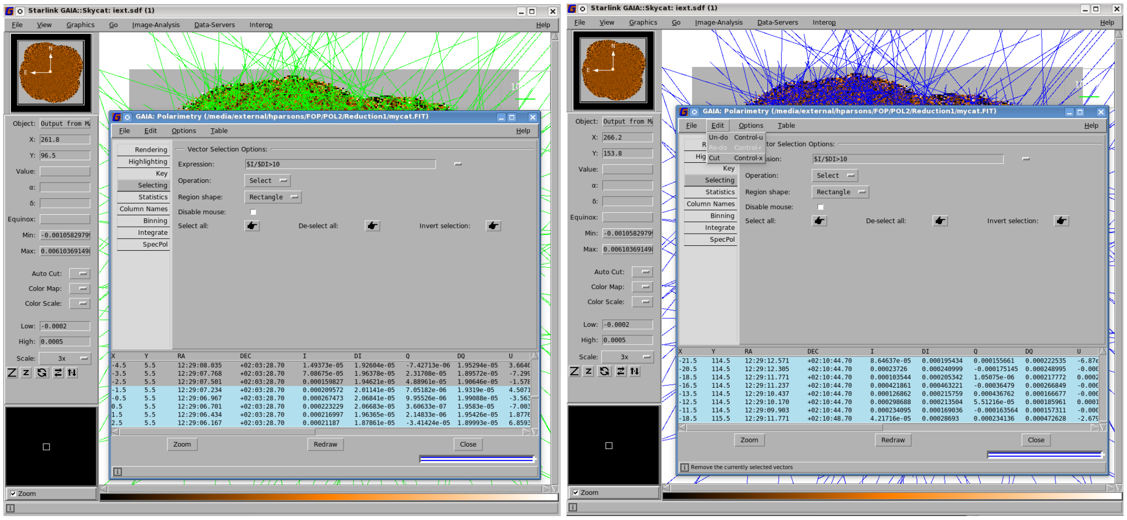


Figure 5.2: Left: specifying vectors to display via the expression `$I/$DI>10`. This will only plot vectors with an intensity signal-to-noise ratio greater than 10 in GAIA. To ensure that this is specified, ensure that "Return" is pressed after entering the expression. Right: After selection, the selected vectors are shown in blue.

Ensure that "Return" is pressed after entering the above expression.

The above expression selects the data points in the polarimetry table that have an associated total intensity (Column **I**) less than 10 times the associated error value for that intensity (Column **DI**). To remove all of these extraneous vectors, either press control-X or use the drop-down menu option **Edit / Cut**. This should leave just a small number of vectors clustering around the target object (see Figure 5.2) [1].

Zooming in on the central region of the map makes it possible to view the vectors near the target object (see Figure 5.3). If needed, it is possible to change the vector scale by selecting the **Rendering** tab in the "`GAIA: Polarimetry`" window.

Finally it is useful for future use (as in the examples in the following sections) to save the final selection of vectors. To save the displayed vectors to a new catalogue, use the drop-down menu **File / Save** in the polarimetry toolbox.

5.2 KAPPA and POLPACK

It is possible to use KAPPA and POLPACK to create POL-2 plots.

```
% kappa
% polpack
```

Note that in the following examples, it will be necessary to ensure that only the vectors to be plotted are included in the file `mycat.FIT`.

There are two main ways to do this — either by saving the output catalogue from GAIA or using the Starlink package POLPACK to manipulate the catalogue produced by the user. To use POLPACK, simply run:

[1]This example is provided here for simplicity. It should be noted, however, that a selection criterion following the general format of `$I/$DI<10 || $AST==0` might actually further improve some selections, as it makes use of the **AST** column to remove vectors outside the `AST` mask (i.e. background vectors).

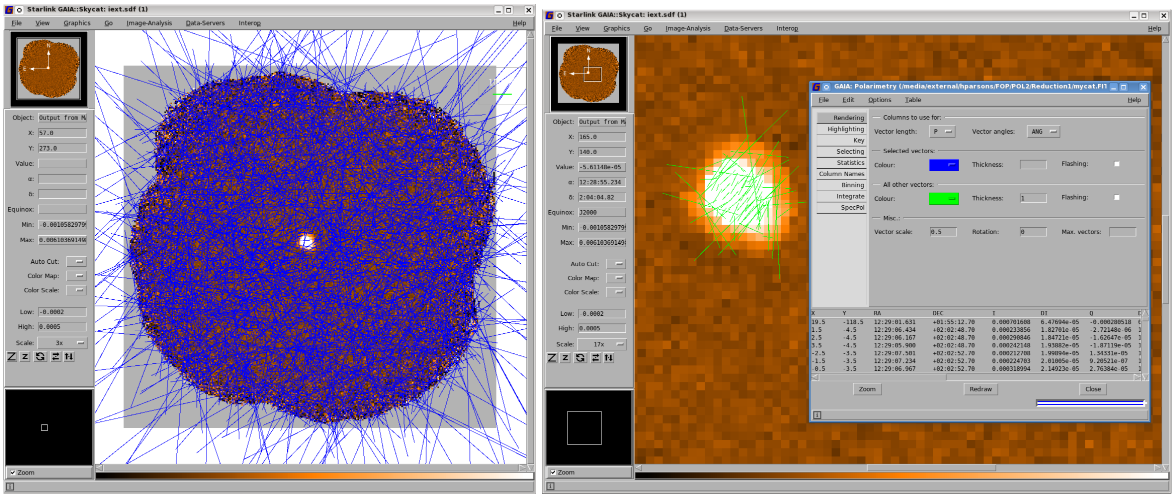


Figure 5.3: Left: selected vectors are marked in blue in this example, Right: after removal of selected vectors, all that remains are the vectors on the (zoomed) regions where $I/$DI > 10.

```
% polpack
```

then select the vectors of interest.

```
% polselect in=mycat.FIT out=selcat.FIT mode=expr "exp='i>10*di && (ast>0)'"
```

The use of the POLPACK polselect command is generally recommended over the use of the CURSA catselect command when working with POL-2 datasets, as it provides a greater number of options specifically designed for use with POL-2 data. For example, polselect offers additional modes that allow the specific selection of vectors that are located within a given region on the sky (specified by an ARD or AST region description), or which correspond to pixels with good values in a specified mask NDF.

Tip

For a better font on PGPLOT PostScript devices, set the following environment variable.

```
% setenv PGPLOT_PS_FONT Times
```

For more info, see `http://pipelinesandarchives.blogspot.com/2015/02/better-fonts-in-postscript-output-from.html`

Graphics-related attributes that can be set are described in SUN95: Descriptions of Plotting Attributes, and the coordinate system attributes that can be set are described in SUN/95: Descriptions of Frame Attributes.

5.2.1 Example 1 – a vector map with no background

In this example, an output file, `plot1.pdf`, is created from the input catalogue `mycat.FIT`.

Select a higher quality PostScript font (Times New Roman in this case):

```
% setenv PGPLOT_PS_FONT Times
```

Select the PostScript graphics device, writing to file `plot1.ps`:

```
gdset plot1.ps/acps
```

For convenience, create a text file holding the main plotting style for polplot:

```
% cat sty
colour=black
drawtitle=0
format(1)=hms
format(2)=dms
```

Likewise, create a text file holding the style for the vector length key:

```
% cat ksty
colour=black
drawtitle=0
```

Plot the vector map. The `vscale` parameter controls the vector scale, and the `keyvec` parameter controls the length of the vector used as the key. There are many other parameters that can be used to control the behaviour of polplot—see the POLPACK manual (SUN/223):

```
% polplot selcat.FIT  style=^sty keystyle=^ksty vscale=20 keyvec=20
```

Convert the map into a PDF file and remove blank margins (if required):

```
% ps2pdf plot1.ps temp.pdf
% pdfcrop temp.pdf plot1.pdf
```

5.2.2 Example 2 – a vector map over a contour map

In this example, an output file, `plot2.pdf`, is created from the input catalogue `mycat.FIT`.

Select the PostScript graphics device, writing to file `plot2.ps`. Note that in this example, a value is not assigned to the `PLOT_PS_FONT` environment variable. This means that the resulting plot uses the default PGPLOT fonts, rather than the higher quality PostScript fonts used in the previous example:

```
% gdset plot2.ps/acps
```

Set up the main plotting style for contour and polplot:

```
% cat sty
colour=black
colour(curves)=red
width(curves)=3
drawtitle=0
format(1)=hms
format(2)=dms
```

Produce the contour map as follows:

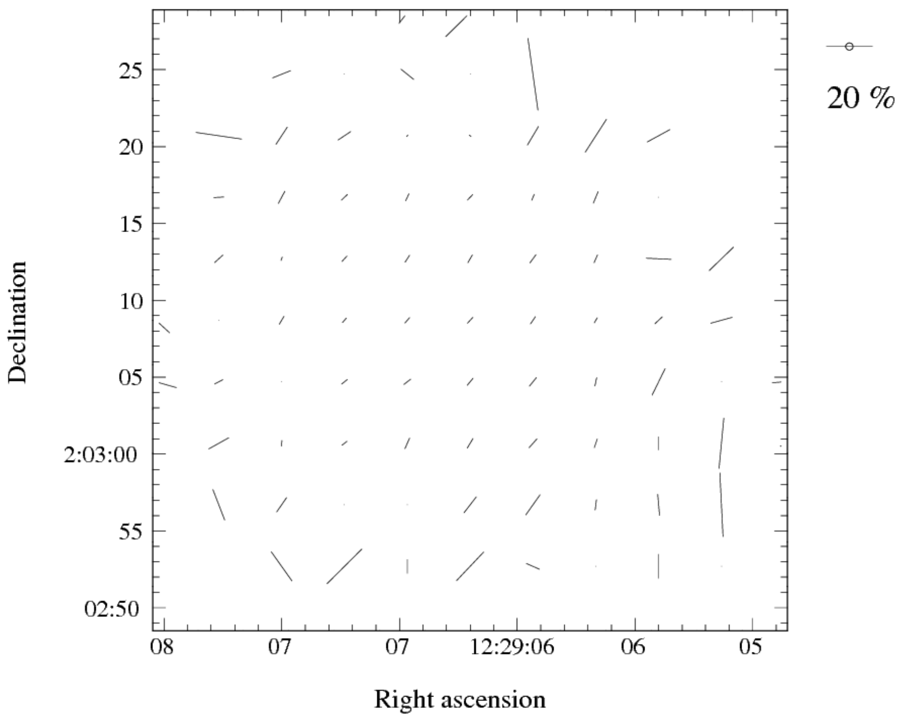


Figure 5.4: Results of Example 1: Producing a vector map with no background using polplot.

```
% contour iext\(0~50,0~50\) mode=perc percentiles=\[88,90,92,94,96,98\] style=^sty key=no
```

Modify the above style for the vector map to produce black vectors:

```
% cat vsty
^sty
colour(curves)=black
```

Set the style for the vector length key:

```
% cat ksty
colour=black
width=3
drawtitle=0
```

Plot the vector map over the contour map. The vectors and contours are aligned automatically in sky coordinates:

```
% polplot selcat.FIT axes=no clear=no style=^vsty keystyle=^ksty vscale=20 keyvec=20
```

Convert into a PDF file and remove blank margins (if required):

```
% ps2pdf plot2.ps temp.pdf
% pdfcrop temp.pdf plot2.pdf
```

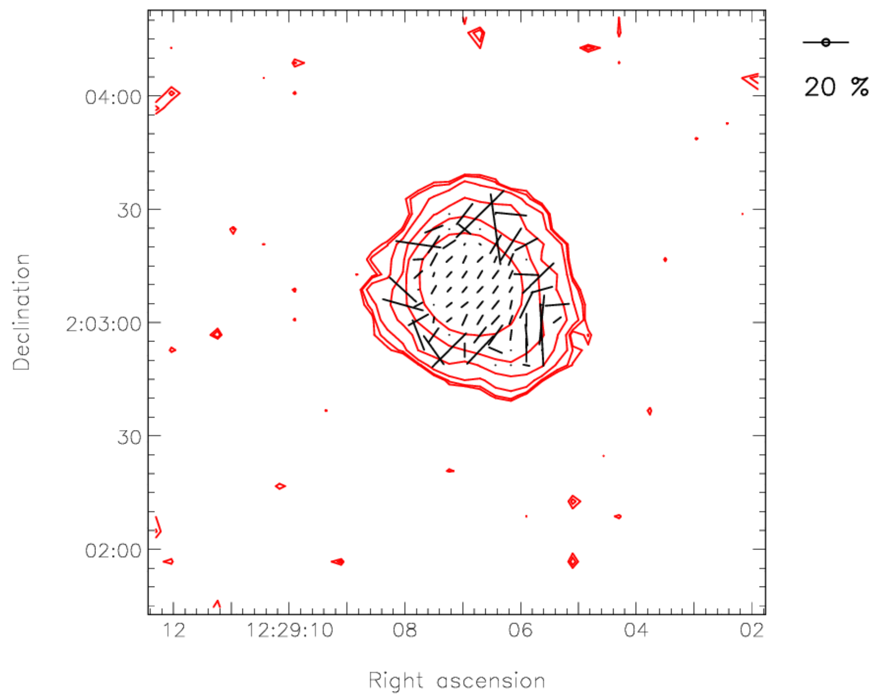


Figure 5.5: Results of Example 2: Producing a vector map over a contour map. This plot uses the default PGPLOT fonts for annotation—note the difference to the fonts used in Figure 5.4.

5.2.3 Example 3 – a vector map over an image

In this example, an output file, `plot3.pdf`, is created from the input catalogue `mycat.FIT`. First, select the appropriate PostScript device (the default PGPLOT fonts are use again, as in the previous example):

```
% gdset plot3.ps/acps
```

To ensure a monochrome colour table is used for the image run lutgrey:

```
% lutgrey
```

Set the main plotting style for display and polplot:

```
% cat sty
colour=black
drawtitle=0
format(1)=hms
format(2)=dms
```

The following function is used to reduce the dynamic range in the map. This allows the user to be able to see structure in the fainter parts, without saturating the brightest regions):

```
maths "'((ia+0.0003)/0.14)**0.2'" ia=iext out=tmp1
```

Display the image, using a reduced range of colours (pens) so that the darkest regions are grey rather than black. This means the black vectors can still be seen within the dark regions:

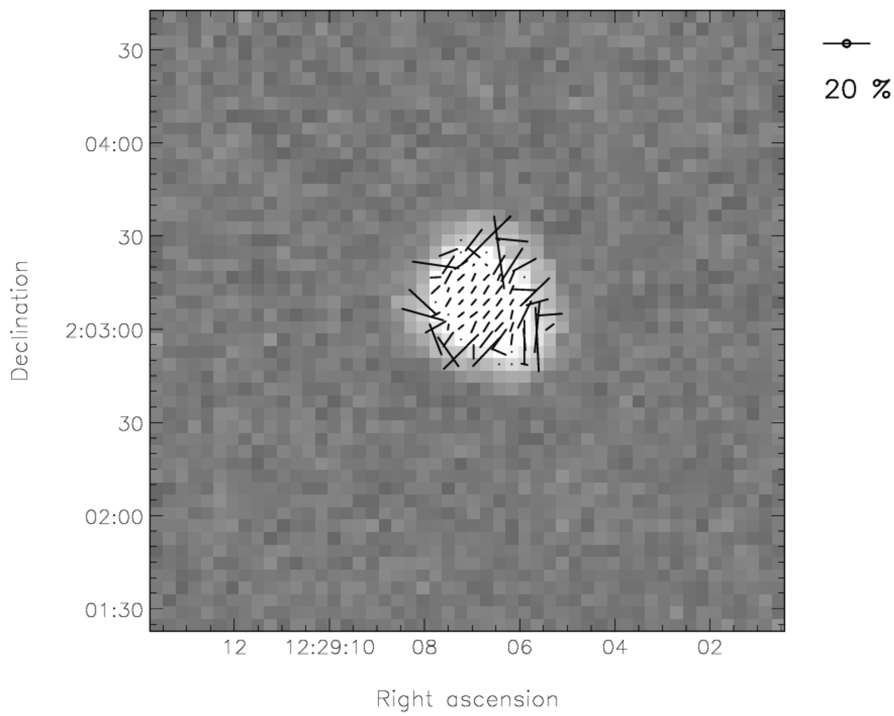


Figure 5.6: Results of Example 3: Producing a vector map over a negative image.

```
% display tmp1\(0~50,0~50\) mode=perc percentiles=\[2,98\] style=^sty \
      penrange=\[0.4,1.0\]
```

Modify the above style for the vector map to produce wider vectors:

```
% cat vsty
^sty
width(curves)=3
```

Set the style for the vector length key:

```
% cat ksty
colour=black
width=3
drawtitle=0
```

Plot the vector map over the contour map. The vectors are aligned automatically with the map:

```
% polplot selcat.FIT axes=no clear=no style=^vsty keystyle=^ksty vscale=20 keyvec=20
```

Convert into a PDF file and remove blank margins (if required):

```
% ps2pdf plot3.ps temp.pdf
% pdfcrop temp.pdf plot3.pdf
```

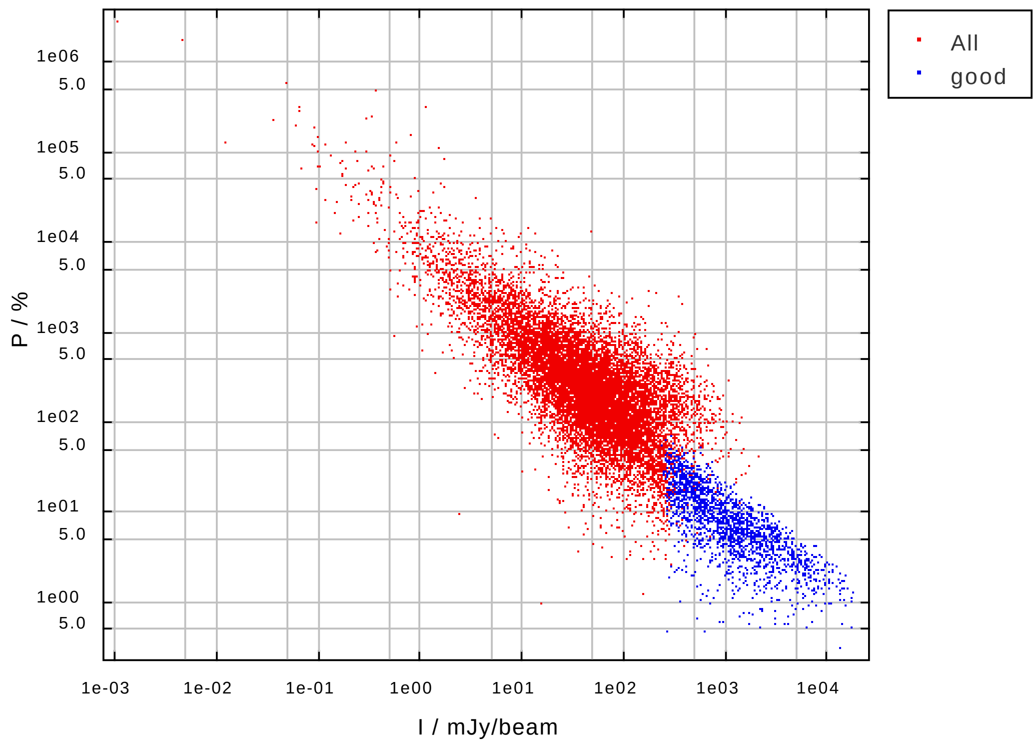


Figure 5.7: A scatter plot of fractional polarisation (P) against total intensity (I) produced by TOPCAT. High signal-to-noise points (I>10.DI) are shown in blue.

5.3 TOPCAT

Catalogues produced by pol2map can be explored using the popular TOPCAT catalogue browser (see `http://www.starlink.ac.uk/topcat/`). For instance:

```
% topcat -f fits mycat.FIT
```

Unlike the other tools described above, TOPCAT cannot produce a visualisation of the catalogue as a set of vectors. However, it goes well beyond the facilities of the other tools in allowing the user to explore correlations between different quantities in the catalogue via two- and three-dimensional scatter plots—see Figure 5.7. It can also be used to cross-correlate two different catalogues, create subsets of a catalogue, create new columns containing related quantities, *etc*. It also allows the modified catalogue to be saved to a new catalogue file on disk. However, users should be aware that any such new catalogue will not contain the WCS information required by other Starlink applications to perform WCS-related operations, such as displaying annotated axes and aligning data-sets. This WCS information can be copied back into the new catalogue, however, using the polwcscopy command (Section 6.4).

Chapter 6
POL-2 – Advanced Data Reduction

The pol2map tool for reducing POL-2 data was originally released for general science community use several years ago. The fact that its development remains ongoing directly reflects the continuing advances made at the cutting edge of POL-2 data reduction and analysis.

This advanced section of the POL-2 data reduction documentation aims to provide you with additional tools and options with which to refine your individual POL-2 data-reduction processes.

For further ideas, see Section 3.7.

6.1 Adding new observations

If you receive additional data after an initial POL-2 reduction of a partial dataset, then it is almost always easier (and the process more robust) for you to simply re-run the reduction process ”from scratch” for the whole, augmented dataset. Despite the associated additional processor time cost, therefore, you are generally recommended to adopt this approach, rather than to attempt to combine new observations with pre-existing pol2map reduction products.

For completeness, however, this section describes the six-step process of combining data for one or more new POL-2 observations into existing I, Q, and U maps and vector catalogue created by an earlier run of pol2map.

(1) Create a text file listing all the existing auto-masked I maps for individual observations stored in the directory specified by Parameter `MAPDIR`, and then add in the raw data files for the new observations. The auto-masked I maps have names that end in `_imap.sdf`.

```
% ls maps/*imap.sdf > infiles.list
% ls rawdata/*.sdf  >> infiles.list
```

(2) Create a new auto-masked, co-added I map including the new observation. The calcqu and makemap commands will be run on the new data and the resulting maps combined with the existing maps derived from the older observations to create the new map:

```
% pol2map in=^infiles iout=iauto_new qout=! uout=! mapdir=maps \
     qudir=qudata
```

(3) A decision needs to be taken as to whether to re-create all the externally masked maps using external masks defined by the new auto-masked map. This will be the case if the auto-masked map has been changed significantly by the addition of the new observation. To do this, it is necessary to compare the old and new masks. The old masks should have been created earlier using the `MASKOUT1` and `MASKOUT2` parameters (see Step 3 in Section 3). To create the new masks that would be generated from the new auto-masked map, use this command.

```
% pol2map  in=^infiles iout=! qout=! uout=! mapdir=maps mask=iauto_new \
     maskout1=astmask_new  maskout2=pcamask_new
```

(4) Decide if the addition of the new data has changed the masks significantly. This involves comparing `astmask.sdf` and `astmask_new.sdf` (and also `pcamask.sdf` and `pcamask_new.sdf`).

(5) If the mask has changed significantly and all observations need to be reprocessed using the new mask, or if skyloop is being used, remove the existing externally masked maps so that they will be re-created by the next invocation of pol2map. Note that this will increase the length of time taken by Step 6 enormously.

Ensure that the new auto-masked co-add is used in place of the old one, in order to define any new masks needed in future:

```
% rm mapdir/*Qmap.sdf mapdir/*Umap.sdf mapdir/*Imap.sdf
% mv iauto.sdf iauto_old.sdf
% mv iauto_new.sdf iauto.sdf
```

(6) Re-create the necessary externally masked maps and co-adds, and then create the new vector catalogue:

```
% pol2map in=qudata/\* iout=iext_new qout=! uout=! mapdir=maps \
     mask=iauto
% pol2map in=qudata/\* iout=! qout=qext_new uout=uext_new mapdir=maps \
     mask=iauto ipref=iext_new cat=mycat_new debias=yes
```

6.2 Experimenting with pixel sizes

Currently,the default map pixel size is 4″ at both 850 and 450 µm. The pixel size is controlled by the `PIXSIZE` parameter in the SMURF pol2map command.

```
% pol2map pixsize=12
```

The following four-step example shows how to investigate the impact of changing pixel size. In this example, 12″ pixels and 7″ pixels are compared.

(1) Begin with an auto-masked total-intensity map from the raw data. For instance:

```
% pol2map in=^myfiles.list iout=iauto12 pixsize=12 qout=! uout=! \
     mapdir=maps12 qudir=qudata
```

(2) Create AST and PCA masks with 12″ pixels from the `iauto12.sdf` file.

```
% pol2map in=qudata/\* iout=! qout=! uout=! mapdir=maps12 mask=iauto12 \
     maskout1=astmask12 maskout2=pcamask12
```

(3) Create masks with 7″ pixels by resampling the 12″ masks created at Step 2. This is done using the KAPPA sqorst command:

```
% sqorst  mode=pixelscale pixscale=\'7,7,7E-05\' in=astmask12 out=astmask7
% sqorst  mode=pixelscale pixscale=\'7,7,7E-05\' in=pcamask12 out=pcamask7
```

(4) Create the 7″ externally masked I, Q, and U maps using the above 7″ masks (note the `mask` parameter value is enclosed in single *and* double quotes).

```
% pol2map in=qudata/\* iout=iext7 qout=qext7 uout=uext7 masktype=mask \
                  mask="'astmask7,pcamask7'" mapdir=maps7 ipref=iext7  \
                  cat=cat7 debias=yes
```

Tip

Using larger pixels usually produces slower convergence, and so the above process will take longer than usual—be patient!

Using larger pixels can sometimes encourage smooth blobs and other artificial features to appear in the map. The `iauto12.sdf` file should be examined to check that it does not have such artificial features.

Check the masks (`astmask12.sdf` and `pcamask12.sdf`) to make sure they look reasonable.

It is usually advisable to leave `PIXSIZE` at its default value and instead use the `BINSIZE` parameter to control the bin size in the vector catalogue—see Section 4.6).

6.3 Investigating systematic error in IP

The error on the IP is reported to be of the order of 0.5%. It is possible to investigate the effects of the systematic error in IP by creating maps using the upper and lower limits on the IP value. The makemap configuration parameter called ipoffset can be used to conduct such an investigation. To use it, run pol2map twice, as follows.

```
% pol2map config="ipoffset=-0.25"
% pol2map config="ipoffset=0.25"
```

This will produce maps using the upper and lower IP limits (a range of 0.5%). If pol2map has already been run on POL-2 data, then a file will already exist that was created using the mean IP (the mean IP is used if `ipoffset` is omitted from the configuration value, or the configuration parameter itself is omitted).

6.4 Adding WCS information back into a vector catalogue

Vector catalogues produced by pol2map contain information about World Coordinate Systems (WCS) in two different forms:

(1) The catalogue contains **RA** and **Dec** columns that hold the sky position (FK5, J2000) of each vector, in radians.

(2) The catalogue header contains a Starlink "WCS FrameSet" which defines (amongst other things) the projection from pixel coordinates within the I, Q, and U mosaics, to RA and Dec. This FrameSet is used by Starlink software, together with the pixels coordinates stored in the **X** and **Y** columns, to determine the RA and Dec of each vector. The WCS FrameSet also defines the polarimetric reference direction used by the **Q**, **U**, and **ANG** values. See "Using World Co-ordinate Systems" within SUN/95 (the KAPPA manual) for more information on the ways in which Starlink software handles WCS information.

Starlink software such as POLPACK, KAPPA, and GAIA rely on the WCS FrameSet for all WCS-related operations (drawing annotated axes, aligning data sets, *etc*). Thus, problems are likely to arise if the WCS

FrameSet is removed from the vector catalogue. This could happen if, for instance, inappropriate software is used to process an existing catalogue, creating a new output catalogue. Under such circumstances, the WCS FrameSet might not be copied to the output catalogue, causing subsequent WCS-related operations to fail. It is safe to use POLPACK, KAPPA, GAIA, and CURSA, as all these packages copy the WCS FrameSet to any new output catalogues. Unfortunately, the popular TOPCAT catalogue browser (see `http://www.starlink.ac.uk/topcat/`) and the STILTS package (`http://www.starlink.ac.uk/stilts/`) upon which it is based, do *not* copy the WCS FrameSet to any output catalogues.

For this reason, POLPACK contains a command that can be used to copy the WCS FrameSet from one catalogue to another. For example: a user creates the catalogue `mycat.FIT` using pol2map, uses TOPCAT to remove low signal-to-noise vectors, and then saves the results to a new catalogue called `selcat.FIT`. The WCS FrameSet would then be missing from `selcat.FIT`, and so it would be necessary to copy it back into place again from the original catalogue file, `mycat.FIT`. To do this, run the "polwcscopy" command.

```
% polwcscopy in=selcat ref=mycat out=selcat2
```

This would create a third catalogue `selcat2.FIT`, which would be a copy of `selcat.FIT`, but with the WCS information inherited from `mycat.FIT`.

6.5 Re-modelling the error estimates in a POL-2 vector catalogue

If the `MAPVAR=YES` option is used when running the SMURF pol2map script, the **I**, **Q**, and **U** error estimates in the resulting vector catalogue will be based on the spread of pixel values at each point on the sky in the I, Q, and U maps made from individual observations. Thus, if 20 observations are processed by pol2map to create a vector catalogue, then each **DI**, **DQ**, or **DU** error estimate in the vector catalogue will be based on the spread of 20 independent measurements of I, Q, or U. Even though 20 observations is a lot of POL-2 data, 20 is still a fairly small number from which to produce an accurate estimate of the error. Consequently, it is usual to see a large level of random "noise" on the error estimates, as in the following example, which shows the total-intensity (I) error estimates taken from a 12″ vector catalogue near Ophiuchus L 1688 (Figure 6.1).

The uncertainty on the error estimate can cause some vectors that are clearly wrong (e.g. because they are very different from nearby vectors) to have anomalously low error estimates and so to be included in the set of "good" vectors (i.e. vectors that pass some suitable selection criterion based on the noise estimates).

One simple solution to this could be to apply some spatial smoothing to the error estimates. This would be a reasonable thing to do if there were no compact sources in the map. The errors close to a compact source are generally higher than those in a background region because of factors such as pointing errors and calibration errors. These factors cause a compact source to appear slightly different in each observation, and so cause higher error estimates in the vector catalogue. The above error estimates map shows this effect in the higher values at the very centre. Simply smoothing this map would spread that central feature out, artificially decreasing the peak error and increasing the errors in the neighbouring background pixels.

An alternative to smoothing is to split the total noise up into several different components, create a model of each component, and then add the models together. The pol2noise script in SMURF enables the user to re-model the noise estimates in a vector catalogue using such an approach. This facility is used by setting `MODE=REMODEL` on the pol2noise command line:

```
% pol2noise mycat.FIT mode=remodel out=newcat.FIT exptime=iext debiastype=mas
```

This creates an output catalogue (`newcat.FIT`) holding a copy of the input catalogue (`mycat.FIT`), and then calculates new values for all the error columns in the output catalogue. The new I, Q, and U error

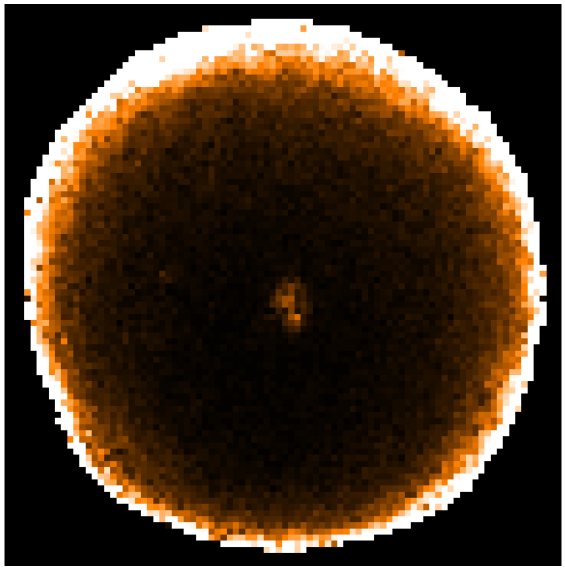


Figure 6.1: The initial total-intensity (I) error estimates taken from a 12″ vector catalogue near Ophiuchus L 1688. Note that the noise level increases towards the edge of the map due to there being fewer bolometer samples per pixel near the edge.

values are first derived from a three-component model of the noise in each quantity, and then errors for the derived quantities (**PI**, **P**, and **ANG**) are found. New values of **PI** and **P** are also found using the specified de-biasing algorithm. The file `iext.sdf` holds the total-intensity co-add map created by pol2map and is used to define the total exposure time in each pixel. The re-modelled total-intensity error estimates are shown in (Figure 6.2).

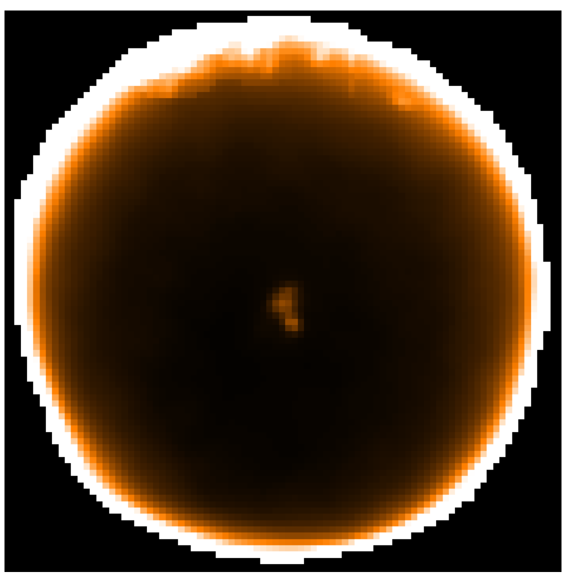


Figure 6.2: The re-modelled total-intensity (I) error estimates taken from a 12″ vector catalogue near Ophiuchus L 1688. Note the reduced noise in comparison with Figure 6.1.

Most of the noise has gone without reducing the resolution. The script displays the original and re-modelled error estimates for each Stokes parameter (I, Q, and U), the residuals between the two, and a scatter plot (Figure 6.3). The best-fitting straight lines through each scatter plot are also displayed.

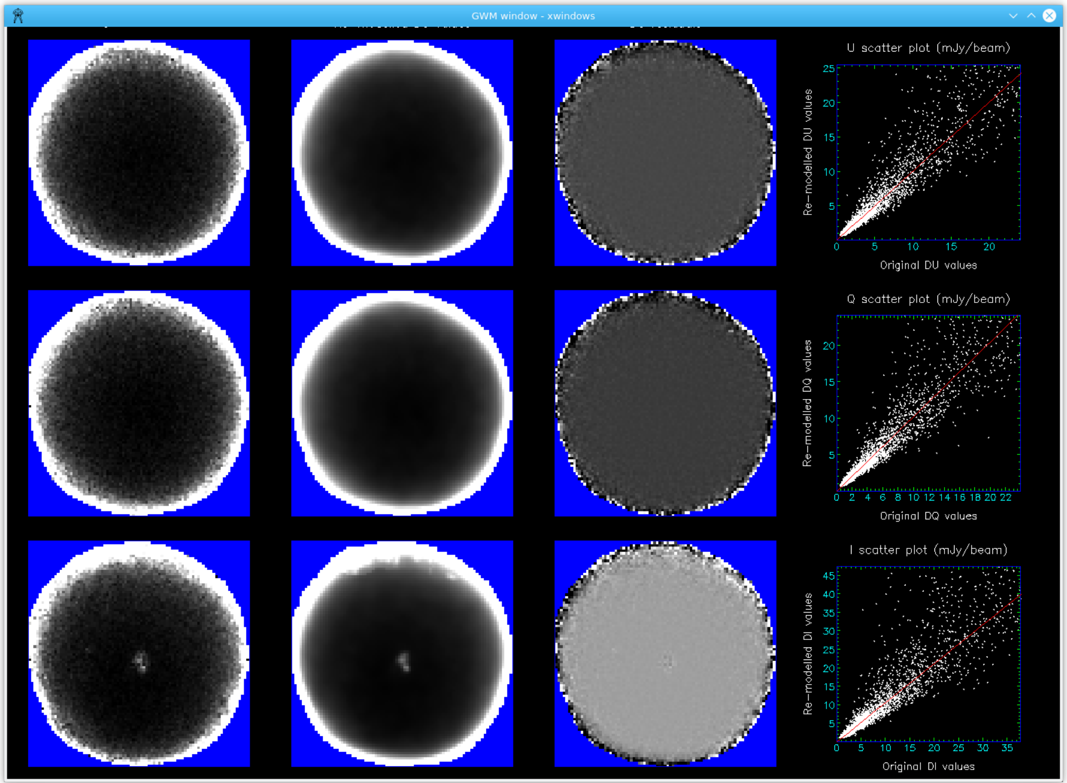


Figure 6.3: Example of the output displayed during the POL-2 vector catalogue error estimate re-modelling process. Comparison Stokes parameter (I, Q, and U) plots are shown, together with residuals images and associated straight-line fits.

The three components used to model the error on each Stokes parameter(I, V, or U) are described below.

(1) **The background component:** This is derived from an exposure-time map (obtained from `iext.sdf` in the above example). The background component is equal to $A.tB$, where t is the exposure time at each pixel, and A and B are constants determined by doing a linear fit between the log of the noise estimate in the catalogue (**DQ**, **DU**, or **DI**) and the log of the exposure time (in practice, B is usually close to -0.5). The fit excludes bright source areas, but also excludes a thin rim around the edge of the map where the original noise estimates are subject to large inaccuracies. Since the exposure-time map is usually very much smoother than the original noise estimates, the background component is also much smoother.

(2) **The source component:** This represents the extra noise found in and around compact sources caused by pointing errors, calibration errors, etc. The background component is first subtracted from the catalogue noise estimates and the residual noise values are then modelled using a collection of Gaussians. This modelling is done using the `GaussClumps` algorithm provided by the findclumps command in the Starlink CUPID package. The noise residuals are first divided into a number of "islands", each island being a collection of contiguous pixels with noise residual significantly higher than zero (this is done using the `FellWalker` algorithm in CUPID). The `GaussClumps` algorithm is then used to model the noise residuals in each island. The resulting model is smoothed lightly using a Gaussian kernel of FWHM 1.2 pixels.

(3) **The residual component:** This represents any noise not accounted for by the other two models. The noise residuals are first found by subtracting the other two components from the original

catalogue noise estimates. Any strong outlier values are removed and the results are smoothed more heavily using a Gaussian kernel of FWHM 4 pixels.

The final model is the sum of the above three components. The new **DI**, **DQ**, and **DU** values are found independently using the above method. The errors for the derived quantities (**DPI**, **DP**, and **DANG**) are then found from **DQ**, **DU**, and **DI** using the usual error-propagation formulae. Finally new **P** and **PI** values are found using a specified form of de-biasing (controlled by the Parameter `DEBIASTYPE`).

6.6 Changing the de-biasing in a POL-2 vector catalogue

The new poledit command in the Starlink POLPACK package has an option to recalculate the **PI** (polarised intensity) and **P** (percentage polarisation) values using a specified form of de-biasing. If the existing catalogue is in file `mycat.FIT`, this can be done with the following commands.

```
% polpack
% poledit mycat newcat mode=debias debiastype=mas
```

This will create a new file, `newcat.FIT`, containing a copy of `mycat.FIT`, but with new **P** and **PI** columns re-calculated from the existing **Q**, **U**, **I**, and **DPI** values using the "Modified Asymptotic" (MAS) bias estimator (any de-biasing used to create the original catalogue is ignored). The options for the `DEBIASTYPE` parameter are:

(1) **MAS:** de-bias using the modified asymptotic estimator;

(2) **AS:** de-bias using the asymptotic estimator; and

(3) **None:** do not include any de-biasing.

Figure 6.4 shows the results of using the poledit command with the various `DEBIASTYPE` parameter options listed above.

6.7 Smoothing 450-µm POL-2 maps to 850-µm-map resolution

The pol2map script has a parameter, `SMOOTH450`, to enable the user to smooth 450 µm POL-2 maps to the same resolution as their 850 µm counterparts. It is only accessed when processing 450 µm data, and defaults to `False`, which results in no changes to the behaviour of pol2map. If `SMOOTH450` is set to `True`, pol2map performs an additional smoothing that results in the resolution of the resulting I, Q, and U co-adds (and the vector catalogue) being degraded to the resolution expected for 850 µm data. This allows 450 and 850 µm results to be compared more easily. A side-effect of the smoothing is that the noise levels in the final 450 µm maps and catalogues is lower.

The smoothing is applied to the I, Q, and U maps made from each individual observation before using them to form the I, Q, and U co-adds. The same smoothing kernel is used for I, Q, and U and is formed by deconvolving the expected 850 µm beam shape using the expected 450 µm beam shape as the PSF (Point Spread Function). The "expected" beam shapes are two-component Gaussians as described in [10].

Figure 6.5 shows the effect of the `SMOOTH450` on the mean radial total-intensity profile of 3C 279 determined from 25 POL-2 observations. It can be seen that the smoothed 450 µm curve is very similar to the 850 µm curve.

Figure 6.6 shows an example of the effect of the `SMOOTH450` on a vector map for POL-2 observations of the molecular cloud DR 21. The main difference is that there are many more red vectors than blue vectors. This is because the smoothing introduced by `SMOOTH450` has reduced the noise level, thus allowing more vectors to pass the above selection criterion.

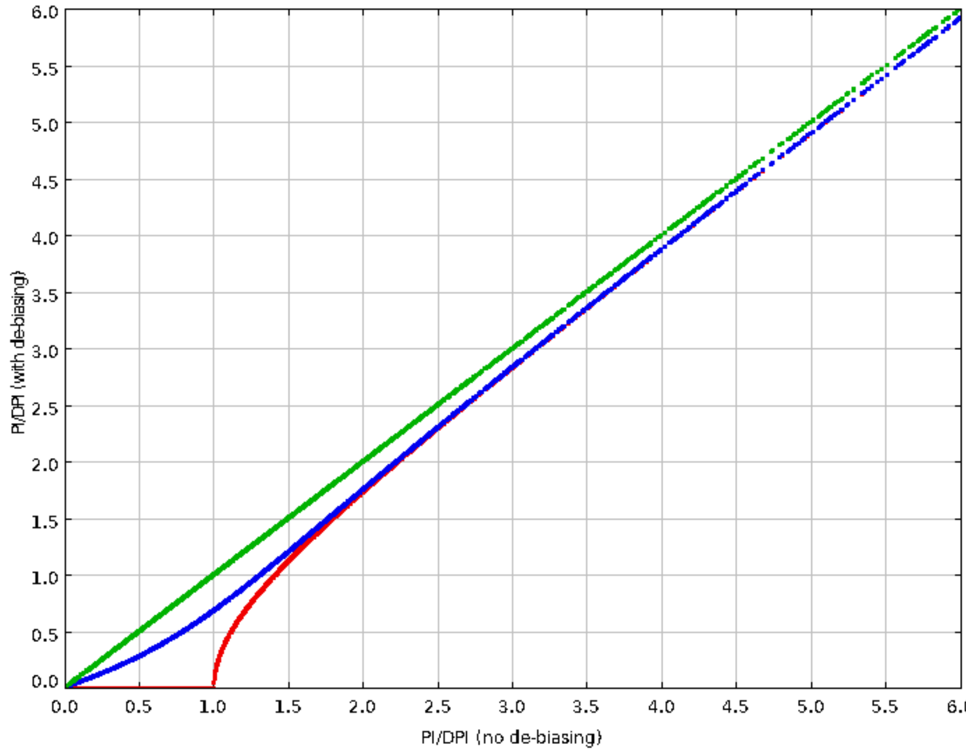


Figure 6.4: The effects of using the various poledit debias options. The horizontal axis is **PI/DPI** (polarised intensity signal-to-noise ratio) with no de-biasing. The vertical axis is the new **PI/DPI** value created with each of the `DEBIASTYPE` options listed above (red is "AS", blue is "MAS", and green is "None").

6.8 Controlling the masks used by pol2map

As noted earlier, the map-making process used by the pol2map command uses two masks, each of which divides the field up into source and background regions.

(1) **The AST mask:** This is used to define the background regions that are to be forced to zero after each iteration of the map-maker algorithm (except the last iteration). This form of masking helps prevent the growth of artificial large-scale structures within the map. Any real astronomical signal present within the masked background regions will tend to be suppressed in the final map, so it is important that the AST mask correctly identifies regions of significant emission down to a low level.

(2) **The PCA mask:** This is used to define the source regions that are to be excluded from the Principal Component Analysis. This analysis is used to remove the correlated backgrounds in the bolometer time-stream data. The time-stream data for astronomical sources are not correlated across bolometers, and so tend to disrupt the PCA. For this reason source regions are excluded from the analysis.

Two separate masks are used because experience has shown that disruption of the PCA is caused mainly by the brighter central source regions. Consequently, the source regions within the PCA mask can be smaller than the source regions within the AST mask, as illustrated in Figure 6.7.

Default masks are created automatically by pol2map in a manner specified by the `MASK` parameter.

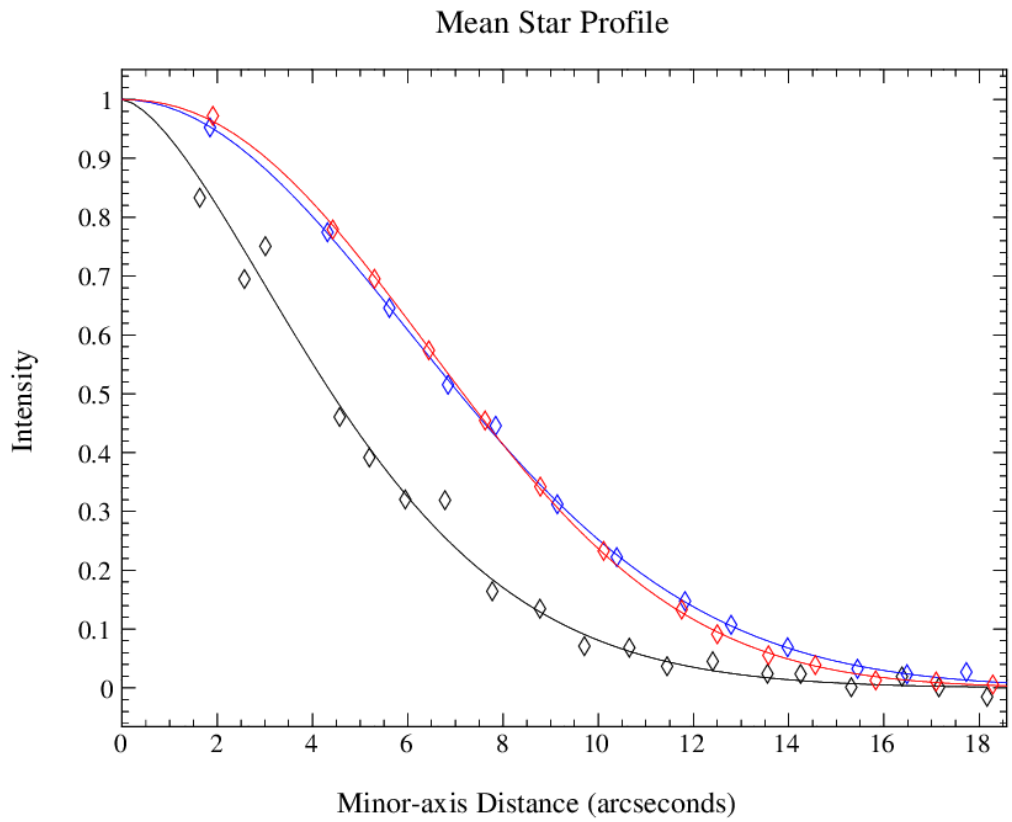


Figure 6.5: The mean radial total-intensity profile of 3C 279 determined from 25 POL-2 observations. The black curve shows the 450 µm profile produced using `SMOOTH450 = NO` (the default), the red curve shows the 450 µm profile produced using `SMOOTH450 = YES` and the blue curve shows the 850 µm profile.

(1) On 'Step 1' of a typical POL-2 data reduction, `MASK` is left at its default value of `Auto`, causing new masks to be generated automatically at the end of each iteration of the map-making algorithm. This "auto-masking" process identifies an initial set of sources by thresholding the current map estimate at the SNR value specified by Configuration Parameter `xxx.ZERO_SNR`, where `xxx` is either `AST` or `PCA`, depending on which mask is being generated. Each of these initial source regions is then expanded to include adjoining pixels down to the SNR level specified by Configuration Parameter `xxx.ZERO_SNRLO`.

(2) On 'Step 2' and 'Step 3' of a typical POL-2 data reduction, `MASK` is set to the co-add of all the total-intensity maps created at Step 1. The pol2map script first creates a pair of AST and PCA masks from the supplied co-added map, and then uses these masks on all iterations of the map-making algorithm [1]. The findclumps command in the Starlink CUPID package is used by pol2map to create the masks. The process used by findclumps is the same as describe above for Step 1–initial sources are defined by a fixed SNR threshold within the supplied co-add and these are then extended down to a lower SNR threshold. The pol2map command sets these threshold values to the values of the four configuration parameters listed above—`AST.ZERO_SNR`, `AST.ZERO_SNRLO`, `PCA.ZERO_SNR`, and

[1] For completeness, it should be mentioned that the COM and FLT models are also masked, in addition to the AST and PCA models. At Step 1 (the auto-masking stage), the masking of COM and FLT is controlled by a similar set of configuration parameters to those of AST or PCA, except that `xxx` becomes `COM` or `FLT`. At Steps 2 and 3 (the external-masking stages), the COM and FLT models are masked using the PCA mask generated by findclumps, and so COM and FLT masking cannot be controlled independently of the PCA mask.

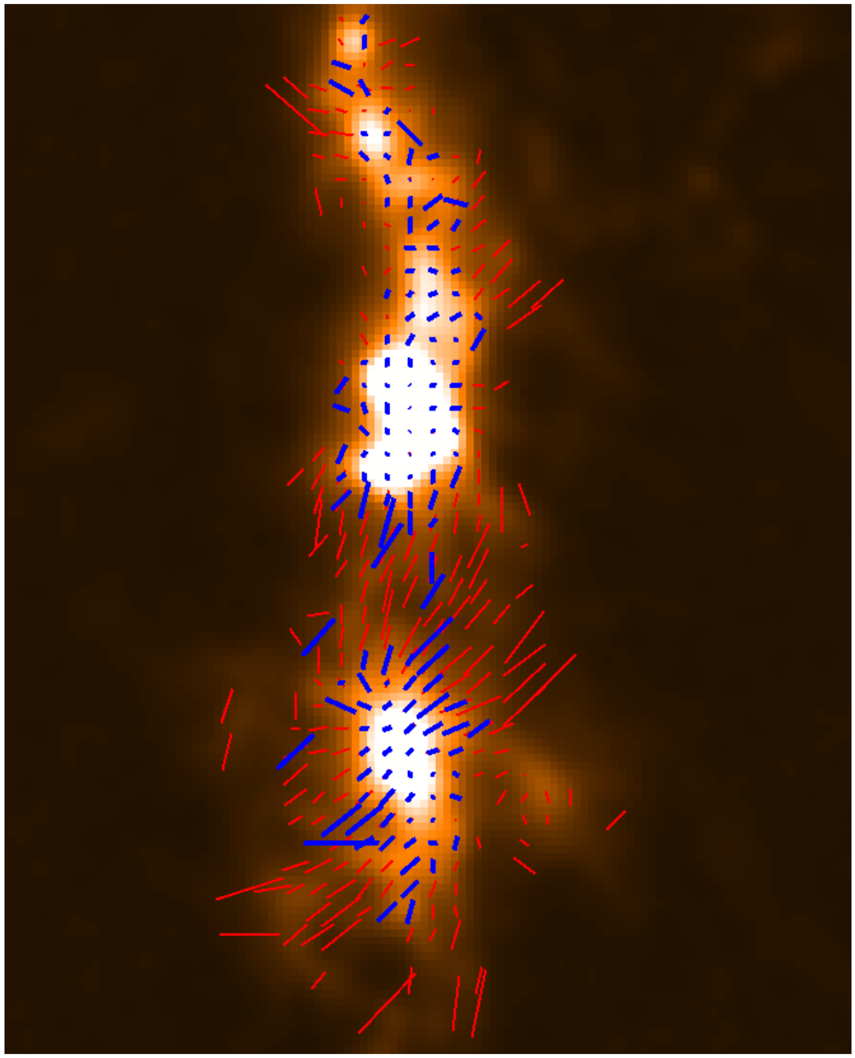


Figure 6.6: 450 µm POL-2 vector maps created from four observations of DR 21, both with and without the use of the SMOOTH450 parameter. The blue vectors were created with SMOOTH450 = NO, and the red vectors with SMOOTH450 = YES. Both red and blue used the same selection criterion (I>5*DI && DP <0.5 && DANG < 10).

PCA.ZERO_SNRLO.

These configuration parameter all default to the following values specified in the pol2map script:

```
AST.ZERO_SNR = 3
AST.ZERO_SNRLO = 2
PCA.ZERO_SNR = 5
PCA.ZERO_SNRLO = 3
```

To investigate the effects of changing these values, new values can be supplied, using the CONFIG parameter of the pol2map command [2]. Here is an example.

[2] Note that prior to 2019 July 10, this method could only be used at Step 1 (the supplied settings were ignored if supplied at Steps 2 or 3). Later versions of pol2map do not suffer from this limitation—the supplied values are honoured at all steps.

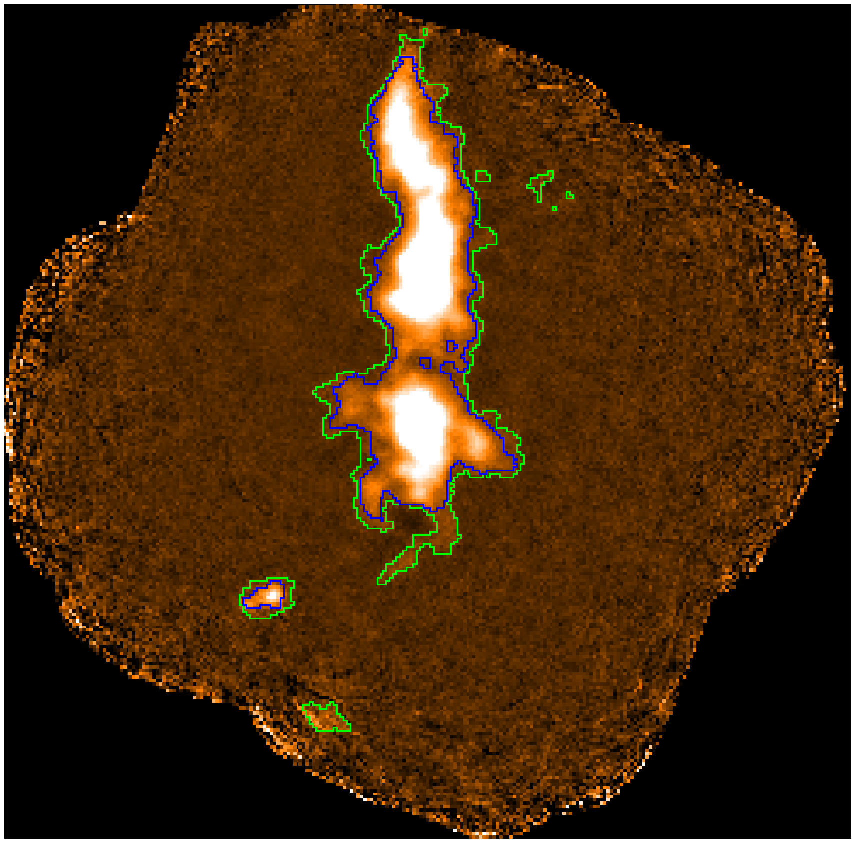


Figure 6.7: A total-intensity map of DR 21, showing the AST mask in green and the slightly smaller PCA mask in blue.

```
% more conf
ast.zero_snr = 2.5
ast.zero_snrlo = 1.5
% pol2map config=^conf
```

Any values not specified will retain their default values listed above.

6.9 Checking for convergence in pol2map log files

When processing POL-2 data, it is important to know that the map-making algorithm converged correctly for all observations. This information is available in the pol2map log file, along with the rest of the makemap or skyloop output. However, the log file can be very long, and so finding the relevant information may not be straightforward, particularly for users unfamiliar with the screen output usually created by makemap or skyloop.

To help with this, a simple Python script called `pol2logcheck.py`[3] is available, which searches a specified pol2map log file for the relevant information and reports any observations that did not converge. This can be used as in the following example.

[3]This script is available at `https://www.eao.hawaii.edu/~dberry/pol2logcheck.py`.

```
% pol2logcheck.py omc1/pol2map.log.3

omc1/pol2map.log.3:

 Looks like a step 1 log file
  The following observation(s) failed to converge:
    20190104 #14
```

6.10 Combining multiple POL-2 fields

To combine POL-2 observations for multiple overlapping fields, the best way to proceed is probably as follows[4].

(1) Run 'Step 1' independently for each field. In other words, use pol2map to create an auto-masked total-intensity map for each field. The following assumes that pol2map is run within directories called `field1`, `field2`, etc., to create the auto-masked maps and the I, Q, and U time-stream data for each field.

(2) Co-add the auto-masked total-intensity maps for all fields. First, create a text file holding the paths to the separate auto-masked total-intensity maps, and then run pol2map as follows.

```
% more infiles
field1/iauto
field2/iauto
% pol2map in=^infiles iout=iauto_mosaic \
          qout=! uout=! multiobject=yes
```

(3) Run 'Step 2' and 'Step 3' for each field, using the mosaic created above as the mask field for all fields. For instance, for the first field:

```
% cd field1
% pol2map in=qudata/\* iout=iext qout=! \
          uout=! mapdir=maps mapvar=yes \
          skyloop=yes mask=../iauto_mosaic
% pol2map in=qudata/\* iout=! qout=qext \
          uout=uext mapdir=maps mapvar=yes \
          skyloop=yes mask=../iauto_mosaic \
          ipref=iext cat=mycat debias=yes
```

(4) then, do the same for `field2`, `field3`, etc. If preferred, Steps 2 and 3 can be combined into a single invocation of pol2map.

```
% cd field1
% pol2map in=qudata/\* iout=iext qout=qext \
          uout=uext mapdir=maps mapvar=yes \
          skyloop=yes mask=../iauto_mosaic \
          ipref=iext cat=mycat debias=yes
```

(5) Co-add the external-masked I, Q, and U maps for all fields and create a vector catalogue from the co-added maps. First, create a text file holding the paths to the external-masked I, Q, and U maps for all fields, and then run pol2map as follows.

[4] This method requires the use of a Starlink build from 2019 June 9 or later.

```
% more infiles
field1/iext
field1/qext
field1/uext
field2/iext
field2/qext
field2/uext
% pol2map in=^infiles iout=iext_mosaic \
          qout=qext_mosaic uout=uext_mosaic \
          multiobject=yes cat=mycat_mosaic \
          debias=yes ipcor=no
```

Bibliography

[1] Archibald, E. N., et al, 2002, *On the atmospheric limitations of ground-based submillimetre astronomy using array receivers*, MNRAS, 336, 1-13 (DOI:10.1046/j.1365-8711.2002.05582.x)

[2] Bastien, P.,et al., 2011, *POL-2: The SCUBA-2 Polarimeter* ASP, 449, 68B 2.1

[3] Berry D. S., 2015, *FellWalker - a Clump Identification Algorithm*, Ast. & Comp., 10, 22-31 (DOI:10.1016/j.ascom.2014.11.004)

[4] Berry D. S, Gledhill T. M, 2015, *POLPACK – An Imaging Polarimetry Reduction Package*, Starlink User Note 223 1.1, 1.3

[5] Cavanagh B., Jenness T., Economou F., Currie M. J., 2008, *The ORAC-DR data reduction pipeline*, Astron. Nactr., 329, 295 (DOI:10.1002/asna.200710944)

[6] Chapin E. L., et al., 2013, *SMURF – Sub-Millimetre User Reduction Facility*, Starlink User Note 258 1.1, 1.3, 1.3.2

[7] Chapin E. L., et al., 2013, *SCUBA-2: iterative map-making with the Sub-Millimetre User Reduction Facility*, MNRAS, 430, 2545 (DOI:10.1093/mnras/stt052)

[8] Currie M. J., Wallace P. T., Warren-Smith R. F., 1989, *Starlink Standard Data Structures*, Starlink General Paper 38.2

[9] Currie M. J., Berry D. S, 2013, *KAPPA – Kernel Application Package*, Starlink User Note 95 1.1, 1.3

[10] Dempsey J. T. et al., 2013, *SCUBA-2: on-sky calibration using submillimetre standard sources*, MNRAS, 430, 2534 (DOI:10.1093/mnras/stt090) 6.7

[11] Dempsey J. T., Friberg P., Jenness T., Bintley D., Holland W. S., 2010 *Extinction correction and on-sky calibration of SCUBA-2*, Proc. SPIE, 7741 (DOI:10.1117/12.856476)

[12] Draper P. W., Gray N., Berry D. S., Taylor M., 2012, *GAIA – Graphical Astronomy and Image Analysis Tool*, Starlink User Note 214 1.1, 1.3

[13] Friberg, P., et al, 2016, *POL-2: a polarimeter for the James-Clerk-Maxwell telescope*, SPIE, Volume 9914, id. 991403 (DOI: 10.1117/12.2231943) 2.1

[14] Gibb A. G., Jenness T., Economou F., 2012, *PICARD — a PIpeline for Combining and Analyzing Reduced Data* Starlink User Note 265

[15] Holland, W. S., et al, 2013, *SCUBA-2: The 10,000 pixel bolometer camera on the James Clerk Maxwell Telescope*, MNRAS, 430, 2513 (DOI:10.1093/mnras/sts612)

[16] Jenness T., et al, 2002, *Towards the automated reduction and calibration of SCUBA data from the James Clerk Maxwell Telescope*, MNRAS, 336, 14-21 (DOI:10.1046/j.1365-8711.2002.05604.x)

[17] Jenness T., et al, 2014, *Learning from 25 years of the extensible N-Dimensional Data Format*, Ast. & Comp., 12, 146, (DOI:10.1016/j.ascom.2014.11.001) 1.3.1

[18] Savini, G., et al. 2009, *Recovering the frequency dependent modulation function of the achromatic half-wave plate for POL-2: the SCUBA-2 polarimeter*, Applied Optics, 48, 2006 (DOI: 10.1364/AO.48.002006) 2.1

[19] Scott D., Van Engelen A., 2005, *Scan Mode Strategies for SCUBA-2*, SCUBA-2 Data Reduction document SC2/ANA/S210/005

Appendix A
PCA on SCUBA-2 data

This document has outlined the process of reducing POL-2 data and its reliance on PCA during the makemap process. There are two main reasons why running makemap with PCA is not the default method for all data observed using SCUBA-2:

(1) it is very time consuming to run because of the faster scanning speed and consequent higher sample rate of non-POL-2 data, and

(2) the optimal number of principal components to remove has not yet been determined.

All the relevant tools are still provided however, and so interested users may wish to try using this method and to compare the results.