SC/19.3

Starlink Project
Starlink Cookbook 19.3

Edward Chapin, Jessica Dempsey, Tim Jenness, Douglas Scott, Holly Thomas & Remo Tilanus

3 August 2010

The SCUBA-2 SRO Data Reduction Cookbook
1.1

Abstract

This cookbook provides a short introduction to Starlink facilities, especially SMURF, the Sub-Millimetre User Reduction Facility, for reducing and displaying SCUBA-2 SRO data. We describe some of the data artefacts present in SCUBA-2 time series and methods we employ to mitigate them. In particular, we illustrate the various steps required to reduce the data, and the Dynamic Iterative Map-Maker, which carries out all of these steps using a single command.

For information on SCUBA-2 data reduction since SRO, please see SC/21.

Contents

List of Figures

1 Introduction

The Submillimetre Common User Bolometer Array-2 (SCUBA-2) is a large-format bolometer camera for the 15-m James Clerk Maxwell Telescope, designed to produce simultaneous continuum images with central wavelengths at 450 and 850 μm.

The purpose of this guide is to help SCUBA-2 users become familiar with the basic facilities for analysing and visualising data using SMURF[1], and the Starlink packages KAPPA[3], and GAIA[4]. Obviously, Starlink must be installed on your system, and Starlink aliases and environment variables must be defined before attempting any of the examples in this document. This guide is *not* aimed at users of the polarimeter (POL-2) or Fourier transform spectrometer (FTS-2).

A brief description of raw SCUBA-2 data is given in Section 2. Section 3 demonstrates how to visualise SCUBA-2 data, interspersed with some simple worked examples of interactive data-reduction techniques. The end goal of this section is *not* to produce a final science-grade map, rather to give the user a feel for the types of artefacts and data reduction steps required to make a useful image. The best way to make an image from SCUBA-2 data is to use the Dynamic Iterative Map-Maker (DIMM). This one-line command and a subset of its control parameters are described in Section 4. We then discuss data calibration (§5) and provide worked examples of cosmology (§6.1) and Galactic (§6.2) data reduction. For the user who wishes only to produce maps in as little time as possible, jump straight to these later sections.

Note that a number of the examples in this document use real data that are distributed with this Starlink release. The small Uranus example is distributed as part of SMURF and can be found in:

```
$STARLINK_DIR/share/smurf/s4a20091214_00015_000*.sdf
```

The more detailed examples use larger data sets which, because of their size, can be downloaded separately via the URL http://www.starlink.ac.uk/extras/sc19/. If you wish to use these data for scientific purposes (i.e. leading to a publication), explicit permission must be obtained from the Director of the James Clerk Maxwell Telescope.

To gain access to SMURF tasks (the data reduction package for SCUBA-2) before proceeding with the examples in this document first type:

```
% smurf

        SMURF commands are now available -- (Version 1.3.1)

        Type smurfhelp for help on SMURF commands.
        Type 'showme sun258' to browse the hypertext documentation.
        Type 'showme sc19' to view the SCUBA-2 map-making cookbook.
```

For a more detailed description refer to the comprehensive Starlink User Note (**SUN/258**)[1].

[1]currently SUN/258 is not completely up to date

2 SCUBA-2 Data Files

The SCUBA-2 data acquisition (DA) system writes data files every 30 s, one file for each of the 40 × 32 pixel subarrays. In addition, each observation can be preceded or followed by calibration frames such as darks or flatfield ramps. For example, observation 15 on 2009-12-14 of Uranus produced the following 16 files with the 450 µm array that was operational at the time (s4a):

```
-rw-r--r-- 1 echapin software  5976576 Jan 17 08:01 s4a20091214_00015_0001.sdf
-rw-r--r-- 1 echapin software 34945536 Jan 17 08:01 s4a20091214_00015_0002.sdf
-rw-r--r-- 1 echapin software 34945536 Jan 17 08:01 s4a20091214_00015_0003.sdf
-rw-r--r-- 1 echapin software 34945536 Jan 17 08:01 s4a20091214_00015_0004.sdf
-rw-r--r-- 1 echapin software 34945536 Jan 17 08:02 s4a20091214_00015_0005.sdf
-rw-r--r-- 1 echapin software 34945536 Jan 17 08:02 s4a20091214_00015_0006.sdf
-rw-r--r-- 1 echapin software 34945536 Jan 17 08:02 s4a20091214_00015_0007.sdf
-rw-r--r-- 1 echapin software 34945536 Jan 17 08:02 s4a20091214_00015_0008.sdf
-rw-r--r-- 1 echapin software 34945536 Jan 17 08:02 s4a20091214_00015_0009.sdf
-rw-r--r-- 1 echapin software 34945536 Jan 17 08:02 s4a20091214_00015_0010.sdf
-rw-r--r-- 1 echapin software 34945536 Jan 17 08:02 s4a20091214_00015_0011.sdf
-rw-r--r-- 1 echapin software 34945536 Jan 17 08:02 s4a20091214_00015_0012.sdf
-rw-r--r-- 1 echapin software 34945536 Jan 17 08:02 s4a20091214_00015_0013.sdf
-rw-r--r-- 1 echapin software 34945536 Jan 17 08:02 s4a20091214_00015_0014.sdf
-rw-r--r-- 1 echapin software 11778560 Jan 17 08:02 s4a20091214_00015_0015.sdf
-rw-r--r-- 1 echapin software  5977600 Jan 17 08:02 s4a20091214_00015_0016.sdf
```

The fifth column gives the file sizes in bytes, and shows that the first and last files, which were dark observations, have nearly identical lengths. Similarly files 2 to 14 also have identical sizes as they each contain precisely 30 s worth of the on-source integration. File 15 is shorter, as the end of the integration lasted less than 30 s. As noted in the introduction, the first two files containing data from the actual scan of Uranus (numbers 2 and 3) are included with this Starlink release.

KAPPA tasks such as fitslist and ndftrace can be used to see the FITS headers and dimensions of the data. In this example, the first and last files are dark observations, and all the other files are produced by a single continuous scan of Uranus. The main data arrays of each file are cubes, with the first two dimensions enumerating columns and rows, and the third time slices (sampled at 200 Hz).

Raw SCUBA-2 data are stored as integers (uncalibrated digitized units). The SMURF task flatfield can be used to scale raw data to units proportional to picowatts (pW, as double precision floating points) using the results of measurements of a flatfield calibration. The calibration can be stored internally in the data file (see the HDS extension `MORE.SCUBA2.FLATCAL`) or can be calculated dynamically from the flatfield ramps bracketing each science observation. The DIMM will always use a flatfield ramp if available. For example, use the command

```
% flatfield 's4a20091214_00015_*.sdf' '*_flat'
```

to produce flat-fielded versions of the files that contain bolometer data taken during the scan across Uranus, with the dark observations (1 and 16 in this example) automatically filtered out.

Other non-science files will also be ignored[2]. Note that the single quotes around the wildcards for the input and output files are necessary since Starlink routines expand them internally, rather than using the shell. However, it is not generally necessary to use flatfield before proceeding with map-making. SMURF will flatfield the data internally by default.

3 Visualising data

In this section several procedures are described for looking at SCUBA-2 data, as well as basic data reduction steps that can be run separately. Working through these examples will illustrate some of the features of SCUBA-2 data, but will *not* result in a science-grade image at the end. If you are interested only in making the best possible map with minimal effort proceed to Section 4.

3.1 Concatenating data

Since SCUBA-2 data for a given subarray are broken into many pieces by the DA system, it is useful for visualisation to first concatenate the data into single files. The SMURF task sc2concat can be used for this operation. For example, assuming the following files are in the current working directory,

```
% sc2concat 's4a20091214_00015_00??.sdf' './*_con'
```

combines all of the files associated with observation 15 for the s4a array into a single file called s4a20091214_00015_0002_con.sdf. sc2concat will automatically filter out any dark observations, so that the concatenated file contains only the data taken during the scan across Uranus with the shutter open. It also applies the flatfield by default, although it can be disabled using the 'noflat' option on the command-line. Be careful when concatenating a very long observation since the output file may be too large to handle sanely. Fifteen minute chunks (30 files) should be more than big enough.

3.2 Displaying scan patterns

The pointing of the telescope throughout a scan (as well as other state information) is stored in the `MORE.SMURF.JCMTSTATE` extension of a data file. The SMURF task jcmtstate2cat will convert this information into a simple ASCII tab separated table:

```
% jcmtstate2cat s4a20091214_00015_0002_con.sdf > state.tst
```

The '-h' option to jcmtstate2cat can be used to find more information on the command. In particular, multiple files can be supplied to the command using standard shell wild cards (not escaped) and for SCUBA-2 data the '–with-mce' option can be used to dump the low-level MCE header information.

This catalogue can be loaded into TOPCAT for plotting, making sure that TOPCAT is told that the TST format is to be used for reading.

[2]Use the SEQ_TYPE FITS header to determine whether a file is part of the main data acquisition sequence or a support file

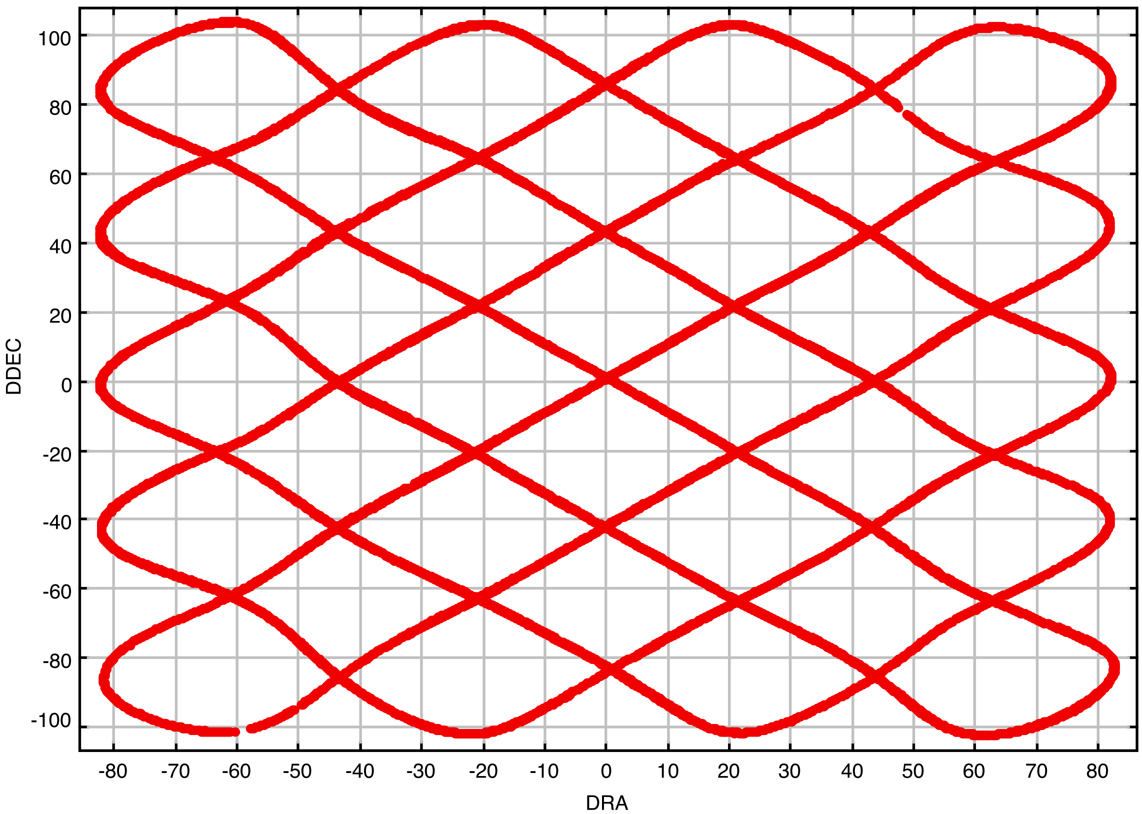

Figure 1: The telescope positions during observation number 7 on 20090107. The plot is created by TOPCAT from the output from jcmtstate2cat plotting the DRA column (right ascension offset from the map centre in arcsec) against the DDEC column (declination offset from the map centre in arcsec). The PONG scan pattern is clearly visible (see Scott & Engelen 2008[5]).

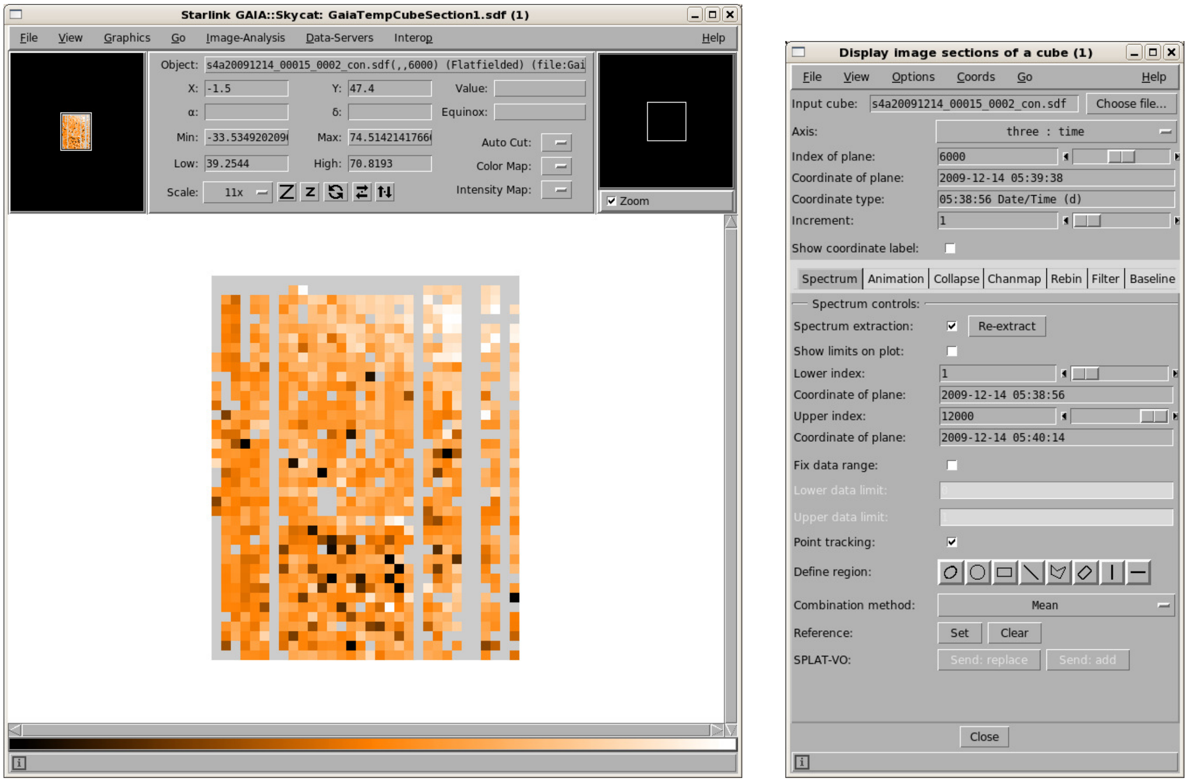

Figure 2: Initial GAIA windows displayed upon loading a data cube (with slight modifications to the colour table using options under the 'File' and then 'Startup options'). **Left:** The main window, after clicking the 'Z' button a number of times to zoom-in, shows a map of bolometer values at a fixed sample in time. Note that these data came from an array with a number of broken columns and broken isolated bolometers, all indicated in grey. **Right:** The 'Display image sections of a cube' dialogue enables the user to navigate the time dimension. The 'Index of plane' slider near the top can be used to select different time slices, and the main window will automatically update.

```
% topcat -f tst state.tst
```

An example plot of the scan pattern for this observation generated by TOPCAT can be seen in Fig. 1. All the time varying header values are available for plotting. In particular DRA and DDEC will show the RA/Dec offset of the telescope, DAZ and DEL will show the Az/El offset and the 225 GHz opacity values are also calculated from the raw WVM measurements.

3.3 Displaying data cubes

The easiest way to visualise the bolometer time series data is to use GAIA. Loading in the concatenated file above (combining the two example files included with this Starlink release) produces two windows (Fig. 2). The main window shows a map of bolometer values at a given instant in time. The second window can be used to navigate the time axis; by moving the 'Index of plane' slider in the 'Display image sections of a cube' dialogue, different time slices may be selected, with the main GAIA window updating automatically.

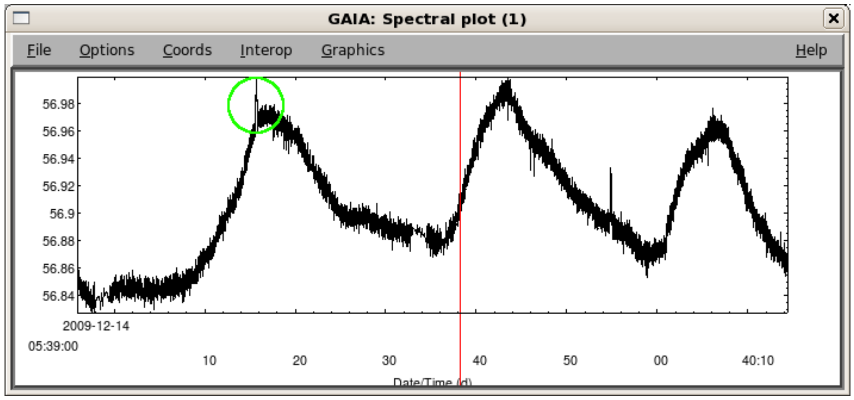

Figure 3: The 'Spectral Plot' window is spawned automatically once a bolometer is clicked in the main window, such as (13,23) in this example, displaying its time-varying signal. The vertical red line indicates the time slice that is currently selected in the 'Display image sections of a cube' dialogue. The regular pattern, with a period of about 30 seconds in this particular case, is slow baseline drift correlated with variations in the SCUBA-2 fridge temperature. The green-circled narrow spike is produced by the bolometer crossing Uranus. Usually astronomical signals are not readily visible in raw time series plots such as these, since they are relatively much fainter. Note that on some systems 'gaps' may appear, such as those at the start of the time series, and at $\sim$ 34 s in this example. These are simply rendering artefacts due to re-sampling the data to the resolution of the display (zooming-in to these regions using the 'Lower index' and 'Upper index' sliders in the 'Display image sections of a cube' dialogue followed by a click on 'Re-extract' to update the plot demonstrates this).

For this concatenated data file (1 minute in total, as it is the combination of two 30 s files), each bolometer has a large offset relative to its neighbours as well as relative to any smaller time-varying signals, which means that little difference can be seen by moving the slider. However, GAIA can produce an automatically scaled plot of the time series for an individual bolometer by simply clicking on it in the main window. For example, clicking on the bolometer at (13,23) spawns the 'Spectral plot'[3] window shown in Fig. 3 for a single bolometer. Clicking on other bolometers over-writes the plot of the original bolometer in the same window. Looking at the vertical axis range on the left, the mean levels clearly vary significantly from bolometer to bolometer, although the time-varying component of the signals are quite similar.

3.4 Regridding data into a map

A simple and quick map can be made from a data cube using the SMURF makemap task. The following will produce a map directly from the raw concatenated data by re-gridding it into a pixelated map:

```
% makemap s4a20091214_00015_0002_con.sdf uranus method=rebin
```

The makemap task automatically scales the bounds of the image to encompass all of the data. The output map here is called '`uranus.sdf`', and the pixel scale is 2 arcsec on a side by default at

[3]This feature of GAIA was originally developed to display spatially-resolved spectra stored in data cubes, hence the name, 'Spectral plot'.

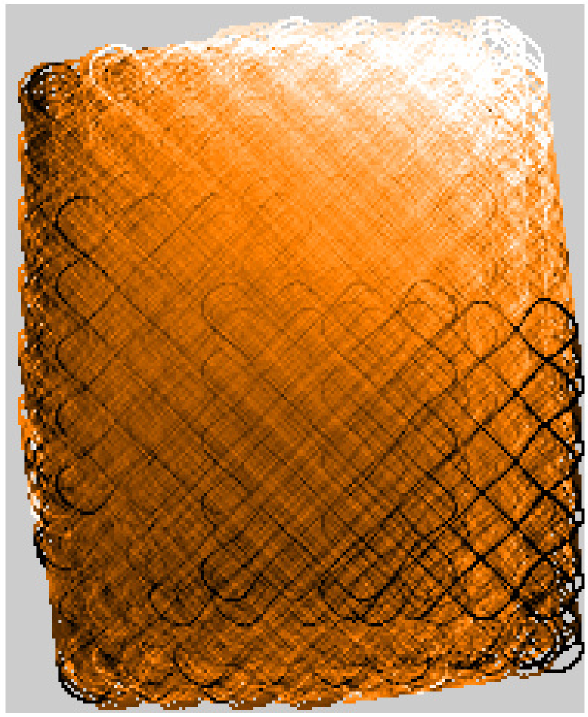

Figure 4: Map produced from raw data of a scan across Uranus. The image is completely dominated by noise in the relative signal offsets of each bolometer, and no astronomical signal can be seen. The scan (a Curvy PONG) can clearly be seen as a repetitive 'waffle' pattern in the image.

450 µm and 4 arcsec at 850 µm (although this can be changed using the 'pixsize=x' option on the command-line, where x is in arcsec)[4]. Since we already know that relative bolometer offsets are large in these data, it is unsurprising that no astronomical source can be seen with GAIA in the resulting image.

3.5 Cleaning data

The previous examples illustrate the need for some kind of data cleaning before there is any hope of seeing astronomical sources. A useful SMURF task for time-domain data processing is sc2clean, which can perform several different steps controlled by a range of parameters. Note that all of the algorithms available to sc2clean are also accessible within the DIMM (Section 4), so in practice the user does not issue this command directly before making the final map.

3.5.1 Removing bolometer averages and DC Steps

The following will remove a 0th-order polynomial (the mean) from each bolometer time stream, storing the cleaned data in a file called 'clean.sdf':

[4]The default sizes are defined as one quarter of the Airy disk rounded up to the nearest half arcsecond.

```
% sc2clean s4a20091214_00015_0002_con.sdf clean \
    config="order=0"
```

This operation results in bolometer data that lie primarily in the range ±0.1 pW, except for a handful of outliers as shown in Fig. 5. In these cases there have been large level changes, or 'DC Steps', during data acquisition. These can be identified and repaired using some extra parameters to sc2clean:

```
% sc2clean s4a20091214_00015_0002_con.sdf clean2 \
  config='"order=0,dcfitbox=30,dcthresh=25,dcsmooth=50"'
```

In addition to removing the mean as in the previous example, the extra parameters starting with 'dc' instruct sc2clean to smooth the data with a median filter of width 50 samples, and then look for steps in the smoothed data in excess of 25-σ. The heights of steps found in this way are measured by fitting straight lines to the smoothed data on either side of the jump using 30 samples in each fit. The corrections thus determined are applied to the original bolometer data. Note that an additional flag has been set by default, `fillgaps=1`, indicating that the data range around the identified steps (±400 samples in this case) should be replaced with a constrained (smooth) realisation of noise to avoid introducing spikes into the data stream (these ranges are ultimately ignored when producing final maps). A map produced from 'clean2.sdf' is shown in Fig. 6.

The command line can get long, so it is also possible to write the configuration parameters into a text file and specify the text file for CONFIG using the standard group notation ('myconfig.lis' in the following example; the leading '^' is necessary, but it is not part of the file name):

```
% sc2clean s4a20091214_00015_0002_con.sdf clean2 \
  config=^myconfig.lis
```

An example config file containing the default values for each item can be found in `$SMURF_DIR/smurf_sc2clean.def`.

3.5.2 Watching a movie

GAIA has the ability to animate the display of a data cube. We will use this feature to make a 'movie' of the array data. Load 'clean2.sdf' from the previous step into GAIA. In the 'Display image sections of a cube' dialogue, switch from the 'Spectrum' to the 'Animation' tab approximately half-way down. Set 'Delay' to 10 milliseconds (the smallest value), 'Step' to 5 (such that it only shows 1 in every 5 frames), and click the 'On' button next to 'Looping'. Finally, click 'Start', and an animation of the data cube will be shown in the main GAIA window. The dominant signal is a gradual variation in the average value of all of the bolometers in unison. This *common-mode* signal is produced through a combination of SCUBA-2 fridge temperature variations, sky noise, telescope motion, and other drifts in the individual bolometers. At times it is also (just barely) possible to see Uranus itself as the array scans across is.

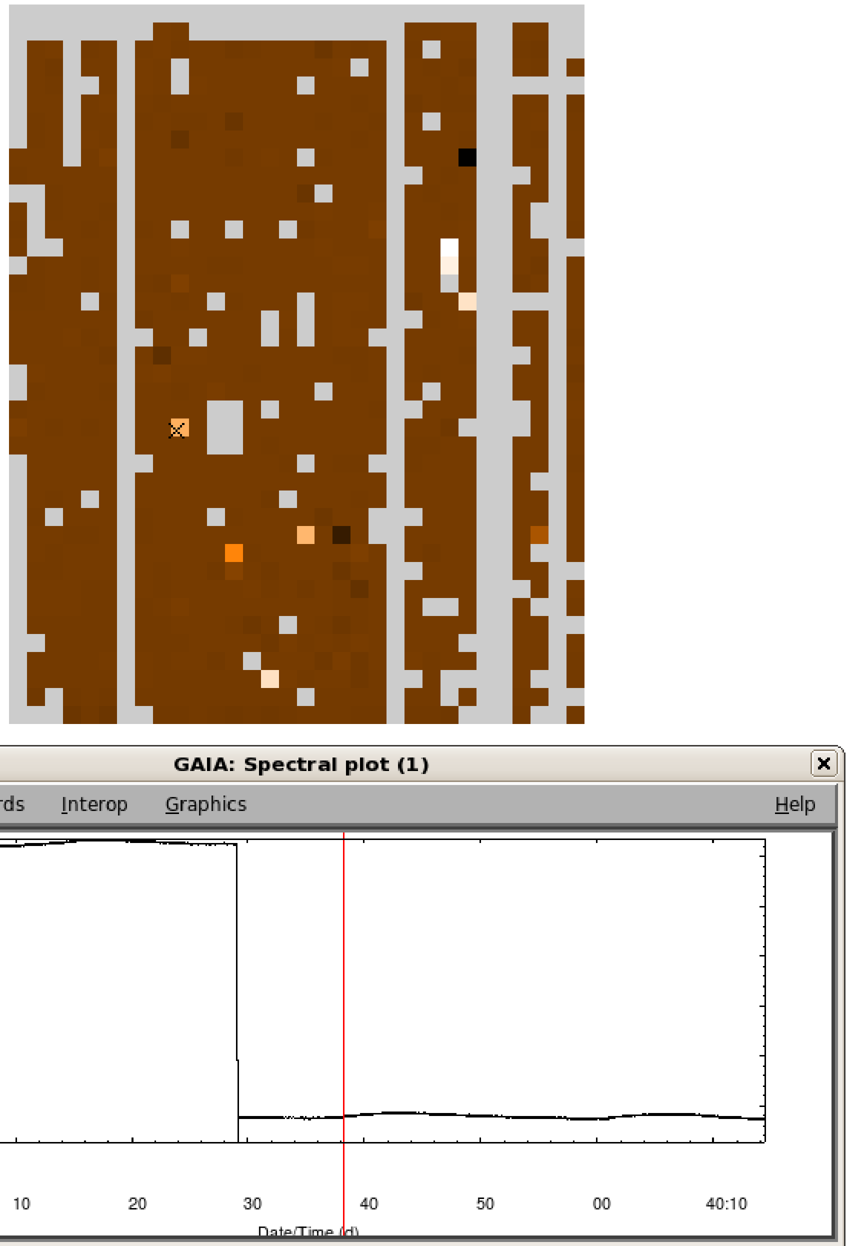

Figure 5: **Top:** GAIA plot of the array signal at time slice 1 after using sc2clean to remove the mean of each bolometer signal (clean.sdf). The LOW and HIGH data values for this intensity plot are −1.7171 pW and +5.50352 pW, respectively. Most of the working bolometers (not grey) now have approximately the same signal range (approximately ±0.1 pW), and therefore have nearly the same colour (brown). However, there are several outliers, for example, the orange pixel with an 'x' at (10,17). **Bottom:** The time series for bolometer (10,17) showing the presence of a large level change, or 'DC Step' near 30 s.

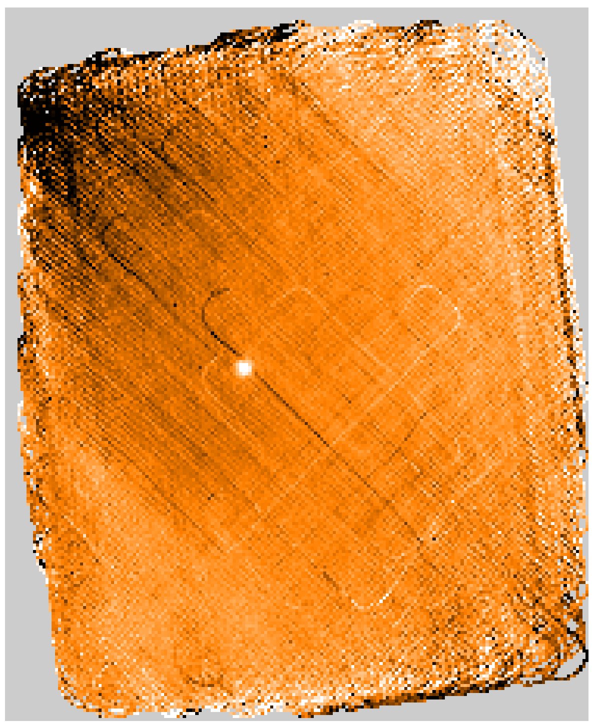

Figure 6: Map produced from 'clean2.sdf' in which DC steps have been repaired and the bolometer means have been subtracted. There are still many artefacts in the data, but now Uranus is visible.

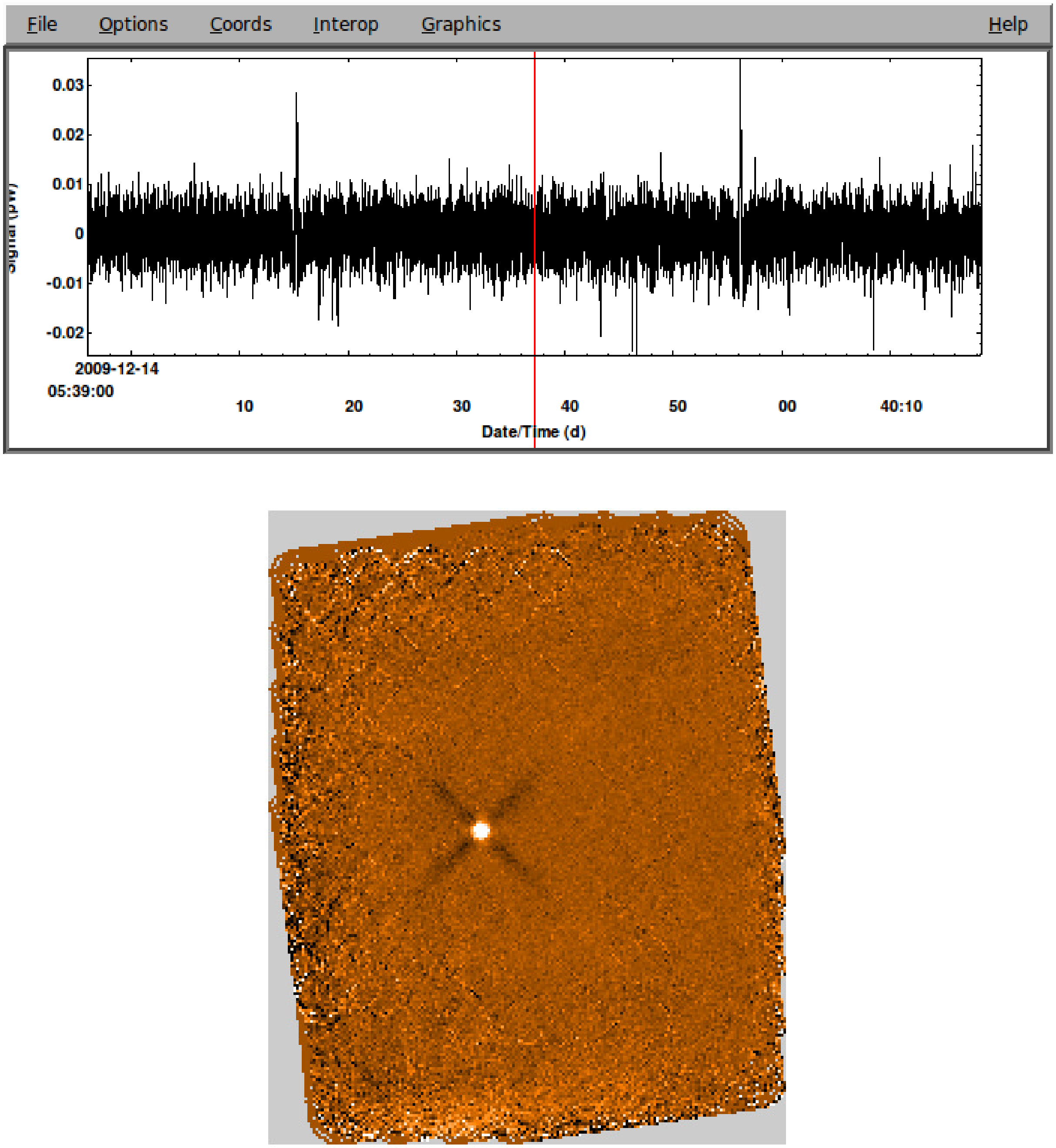

Figure 7: High-pass filtered data (clean3.sdf). **Top:** The time series for a single bolometer (13,23). Comparing with the un-filtered data in Fig. 3, the positive spikes produced by Uranus are now much more apparent. **Bottom:** a map produced from these cleaned and filtered data. The brightness of Uranus compared to the noise is now much more significant than in Fig. 6, but the filtering has caused ringing around the source-crossing in the time series, which appears as negative dips in the map along the scan directions (i.e. a negative cross around the source).

3.5.3 Frequency-domain filtering

sc2clean can also perform frequency domain filtering on the bolometers. In the following example data are concatenated, and filtered using a single call to sc2clean.

```
% sc2clean 's4a20091214_00015_000?.sdf' clean3 \
   config=^${STARLINK_DIR}/share/smurf/sc19_clean3.lis

Processing data from instrument 'SCUBA-2' for object 'URANUS' from the
following observation  :
  20091214 #15 scan
```

Here the text file, `sc19_clean3.lis`, contains the following configuration options:

```
order=0
dcfitbox=30
dcthresh=25
dcsmooth=50
fillgaps=1
apod=<undef>
filt_edgehigh=0.5
```

As in the early examples, the first four parameters cause sc2clean to remove the mean bolometer values, and repair DC steps. Next, the `apod=<undef>` (combined with an internal default `zeropad=0`) is a slightly cryptic way to tell sc2clean to enforce continuity between the starts and ends of each bolometer time-series by temporarily adding padding that is a function of the filter frequency, and interpolating the ends using a cubic function. This step is required to avoid ringing once the FFT is taken. The alternative, and now deprecated, method for reduing ringing is to pad and apodize. Finally, the last parameter tells sc2clean to apply a high-pass filter with a hard lower cutoff frequency of 0.5 Hz.

Fig. 7 shows the filtered bolometer signal for (13,23), as well as a map produced from the data. The filtering has significantly improved the noise properties of the map compared with Fig. 6, but the filtering has now introduced ringing around the source which results in a negative cross pattern along the scan directions.

3.6 Bolometer power-spectra

The frequency-domain power spectra of SCUBA-2 bolometers can be produced with the SMURF task sc2fft. Similar to the previous example, we first produce cleaned, concatenated data. However, we omit the high-pass filtering by overriding the 'filt_edgehigh' parameter explicitly, since the goal now is to see what the full bolometer noise power spectrum looks like.

```
% sc2clean 's4a20091214_00015_000?.sdf' clean4 \
   config="'^${STARLINK_DIR}/share/smurf/sc19_clean3.lis,filt_edgehigh=0'"

Processing data from instrument 'SCUBA-2' for object 'URANUS' from the
following observation  :
```

```
  20091214 #15 scan

% sc2fft clean4 pspec power=true

Processing data from instrument 'SCUBA-2' for object 'URANUS' from the
following observation  :
  20091214 #15 scan

Found 1 continuous chunk
SC2FFT: power spectrum requested so setting POLAR=TRUE
```

While there is a Starlink standard format for storing complex values as described in [2], sc2fft uses its own local format that allows both for Cartesian and polar forms: a 4-dimensional array, with the first axis indicating frequency, the second and third axes bolometer row and column, and the final axis containing the real and imaginary parts of each Fourier coefficient. By setting 'power=true' it switches to polar power form, such that the first element of the 4th array axis stores the square of the amplitudes, and the second element stores the arguments (phases) of each Fourier coefficient. As with the time series data, GAIA may then be used to view the data cube. The difference is that we now specify a subset of the data so that the 3rd axis of the cube is the array of squared Fourier amplitudes for each bolometer, and ignore the phases (i.e. we observe only the 1st elements along the 4th axis):

```
% gaia 'pspec(,,,1)'
```

It is then possible to click on each bolometer to display its power spectrum. Once the 'Spectral plot' window has been spawned, it will also be necessary to modify the axis displays. Select 'Options'→'Positive Y Only', and 'Options'→'Log Y Axis'. Similarly, select the corresponding settings for the 'X' axis. Fig. 8 shows the power spectrum of bolometer (13,23), with a $1/f$ knee apparent close to 1 Hz.

3.7 Data quality flagging

NDF files manipulated by SMURF use the standard Starlink named Quality mechanism (see discussion of Masking, Bad Values, and Quality in SUN/95). Quality itself is stored as an 8-bit integer for each sample in a data file, and each bit can reflect a different condition. For example, the following will indicate the number of samples flagged by sc2clean when producing 'clean4.sdf' in the previous example:

```
% kappa


     KAPPA commands are now available -- (Version 1.13-2)

     Type kaphelp for help on KAPPA commands.
     Type 'showme sun95' to browse the hypertext documentation.

     NOTE, several applications have had major changes made to their
     parameter lists. See the 'Release Notes' section of SUN/95 for
     details.
```

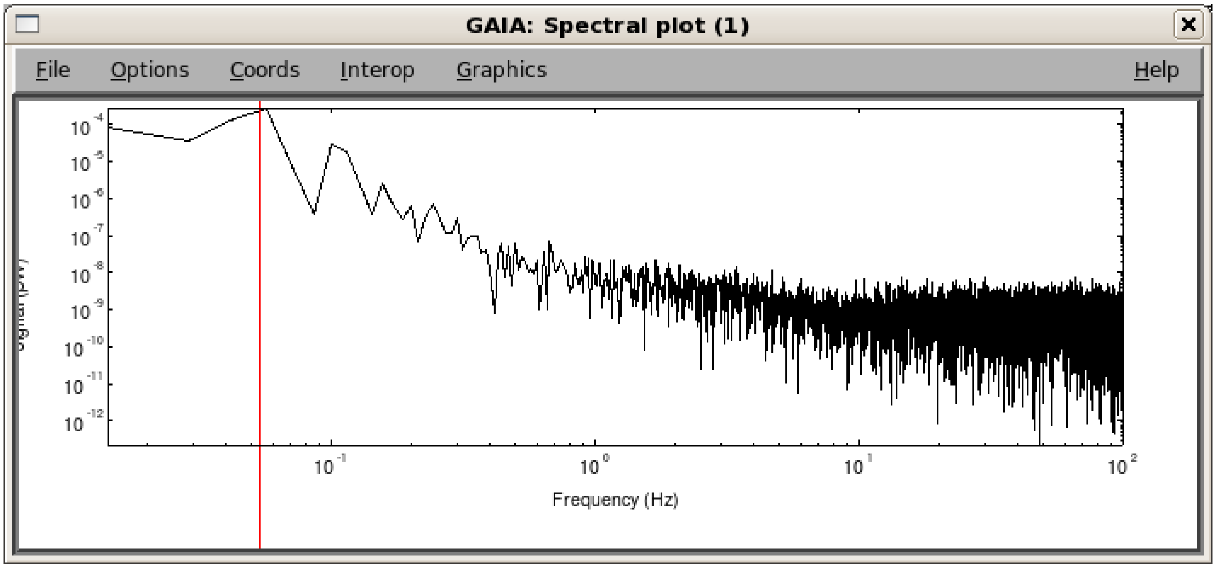

Figure 8: The power spectrum of bolometer (13,23) produced by the SMURF task sc2fft. In this particular case the noise looks flat above a $1/f$ knee of about 1 Hz. The red line simply indicates the frequency slice currently being displayed in the main GAIA window (not shown).

```
% showqual clean4 count
  BADDA           (bit 1) - "Set iff a sample is flagged by the DA" (4416000)
  BADBOL          (bit 2) - "Set iff all data from bolo to be ignored" (4476000)
  DCJUMP          (bit 3) - "Set iff a DC jump is present" (6924)
```

The 'count' supplied to showqual indicates that the total number of occurrences of each Quality flag in the data should be counted and displayed. This example shows that 4416000 samples in total were flagged 'BADDA' by the flatfielding algorithm or by the data acquisition (DA) system itself based on the array setup. Since each bolometer produced 12000 samples, this corresponds to $4416000/12000 = 368$ bolometers that were not operational (out of 1280). The 'BADBOL' flag is used by SMURF to flag *every* sample of a bolometer that is not being used. In this case the number of flags is 4476000 which corresponds to 373 bolometers. In other words, cleaning turned off an additional 5 bolometers. Finally, 'DCJUMP' indicates the number of samples that were affected by the DC step correction procedure (which include a number of samples on either side of the precise locations where the steps occurred).

During map-making flagged samples will not be used and by default they will be hidden from view for all Starlink tools. However, if you wish to see the data that were flagged you can use the KAPPA setbb command to enable a particular flag or turn them all off (so that they can be seen in GAIA).

```
% setbb clean4 0
```

will disable quality and make visible all the underlying data. Using a value of 255 will turn all bits back on and so any non-zero quality will be treated as bad data.

Additionally, the KAPPA task qualtobad may be used to permanently convert samples with given quality bits to a *magic* or *invalid* value. For example:

```
% qualtobad clean4 badmasked DCJUMP
```

will set all of the samples flagged with quality DCJUMP to the magic value. When 'badmasked.sdf' is subsequently viewed with GAIA, none of those portions of the data will be visible.

4 Dynamic Iterative Map-Making

Rather than cleaning the data by hand and then re-gridding it all at the end, we can instead do everything at once using the SMURF Dynamic Iterative Map-Maker (DIMM).

The DIMM is enabled using the method=iterate switch to the makemap task. In the following example we will produce a map of Uranus using the test data supplied with this Starlink distribution. All of the settings for the DIMM are stored in configuration files. In this example we will use one of the examples that are installed in the Starlink tree, 'dimmconfig.lis'. For an overview of the different default configuration files available see Section 4.1. The parameters are also fully described in the makemap documentation[5] or the config files themselves. A local copy can be made and altered if desired. The following command invokes the DIMM to produce the image shown in Fig. 9 with 2 arcsec pixels:

```
% makemap '$STARLINK_DIR/share/smurf/s4a20091214_00015_000?.sdf' uranus 2 \
method=iterate config=^$STARLINK_DIR/share/smurf/dimmconfig.lis

Out of 2 input files, 0 were darks, 0 were fast flats and 2 were science
Processing data from instrument 'SCUBA-2' for object 'URANUS' from the
following observation  :
  20091214 #15 scan

MAKEMAP: Map-maker will use no more than 38670 MiB of memory

   Output sky coordinates are (RA,Dec) offsets from the (moving)
   telescope base position, which started at (RA,Dec) = (23:34:38.6,
-3:33:57).

   Projection parameters used:
      CRPIX1 = 6.95327973673941e-310
      CRPIX2 = 0
      CRVAL1 = 0 ( DRA = 0:00:00.000 )
      CRVAL2 = 0 ( DDec = 0:00:00.00 )
      CDELT1 = -0.000555555555555556 ( -2 arcsec )
      CDELT2 = 0.000555555555555556 ( 2 arcsec )
      CROTA2 = 0

   Output map pixel bounds: ( -74:112, -113:116 )
```

[5]See **SUN/258** or use the smurfhelp command.

```
   Output map WCS bounds:
        Right ascension offset: -0:00:14.867 -> 0:00:10.067
        Declination offset: -0:03:49.00 -> 0:03:51.00

smf_iteratemap: Iterate to convergence (max 5)
smf_iteratemap: Stopping criteria is a change in chi^2 < 0.001
smf_iteratemap: map-making requires 1355 MiB (map=1 MiB model calc=1353 MiB)
smf_iteratemap: Continuous chunk 1 / 1 =========
smf_iteratemap: Iteration 1 / 5 ---------------
--- Size of the entire data array ------------------------------------------
bolos  : 1280
tslices: bnd:0(0.0 min), map:12000(1.4 min), tot:12000(1.4 min)
Total samples: 15360000
--- Quality flagging statistics --------------------------------------------
 BADDA:    4416000 (28.75%),         368 bolos
BADBOL:    4956000 (32.27%),         413 bolos
DCJUMP:       1557 ( 0.01%),
 NOISE:     480000 ( 3.12%),          40 bolos
Total samples available for map:   10402995, 67.73% of max (866.916 bolos)
smf_iteratemap: Calculate time-stream model components
smf_iteratemap: Rebin residual to estimate MAP
smf_iteratemap: Will calculate chi^2 next iteration
smf_iteratemap: Calculate ast
--- Quality flagging statistics --------------------------------------------
 BADDA:    4416000 (28.75%),         368 bolos  ,change          0 (+0.00%)
BADBOL:    5292000 (34.45%),         441 bolos  ,change     336000 (+6.78%)
 SPIKE:        137 ( 0.00%),                    ,change        137 (+inf%)
DCJUMP:       1557 ( 0.01%),                    ,change          0 (+0.00%)
   COM:     369162 ( 2.40%),                    ,change     369162 (+inf%)
 NOISE:     480000 ( 3.12%),          40 bolos  ,change          0 (+0.00%)
Total samples available for map:   10033706, 65.32% of max (836.142 bolos)
     Change from last report:    -369289, -3.55% of previous
smf_iteratemap: Iteration 2 / 5 ---------------
smf_iteratemap: Calculate time-stream model components
smf_iteratemap: Rebin residual to estimate MAP
smf_iteratemap: *** CHISQUARED = 1.15486085552371 for filegroup 1
```

This excerpt shows the initial output of the DIMM. Note that the basic dimensions of the data being processed are listed near the start of the first iteration, as well as the QUALITY flagging statistics. This report is similar to that produced by the KAPPA task showqual in Section 3.7. At the beginning, the main purpose is to indicate how many bolometers are being used: 368 bolometers were turned off during data acquisition (`BADDA`); and 40 bolometers exceeded the acceptable noise threshold (`NOISE`). There were also small numbers of samples flagged as containing spikes and jumps. The total number of bad bolometers (`BADBOL`) is 413. Accounting for these, and the small numbers of additionally flagged samples, 866.916 effective bolometers are available at the start of the first iteration.

After each subsequent iteration a new QUALITY report is produced, indicating how the flags have changed. An important flag that appears in the QUALITY report following the first iteration is `COM`: the DIMM rejects bolometers (or portions of their time series) if they differ significantly from the common-mode (average) of the remaining bolometers. Another useful quantity reported is `CHISQUARED` – the RMS of the residual (time series with the various model components removed) for all of the samples with good QUALITY, normalized by the measured white noise levels.

This particular configuration executes a maximum of 5 iterations, but stops sooner if the change in `CHISQUARED` is less then 0.001. In this case, it reaches this stopping criterion after 5 iterations:

```
smf_iteratemap: Iteration 5 / 5 ---------------
smf_iteratemap: Calculate time-stream model components
smf_iteratemap: Rebin residual to estimate MAP
smf_iteratemap: *** CHISQUARED = 1.14451425189281 for filegroup 1
smf_iteratemap: *** change: -2.48779013769518e-05
smf_iteratemap: Calculate ast
--- Quality flagging statistics --------------------------------------------
 BADDA:    4416000 (28.75%),        368 bolos  ,change          0 (+0.00%)
BADBOL:    5304000 (34.53%),        442 bolos  ,change          0 (+0.00%)
 SPIKE:        180 ( 0.00%),                   ,change          0 (+0.00%)
DCJUMP:       1557 ( 0.01%),                   ,change          0 (+0.00%)
   COM:     392350 ( 2.55%),                   ,change       1670 (+0.43%)
 NOISE:     480000 ( 3.12%),         40 bolos  ,change          0 (+0.00%)
Total samples available for map:   10010475, 65.17% of max (834.206 bolos)
     Change from last report:      -1670, -0.02% of previous
smf_iteratemap: ****** Completed in 5 iterations
smf_iteratemap: ****** Solution CONVERGED
Total samples available from all chunks: 10010475 (834.206 bolos)
```

Note that compared to the initial report, the total number of samples with good QUALITY (`Total samples available for map`) has dropped from 10402995 to 10010475 (about a 4 per cent decrease) as additional samples were flagged in each iteration.

The iterative map-maker estimates several components of the bolometer signal in addition to the astronomical signal that goes into the map. The particular sequence of components that it fits is specified by `modelorder` in the configuration file. All of the examples provided with SMURF are derived from `$STARLINK_DIR/share/smurf/dimmconfig.lis`, and we show the relevant excerpt here:

```
# Model components/order (comma separated list in brackets)
# Note: components specified AFTER 'ast' will not be calculated for the
# first time until the second iteration.
#  dks = fit and remove dark squid for the column
#  com = remove common-mode signal
#  gai = if com specified, fit gain/offset of common mode
#  ext = apply extinction correction
#  ast = estimate the map and astronomical signal
#  flt = apply filter to time streams
#  noi = estimate time-domain variance
#  smo = time series smoothing using a median or mean boxcar filter
#  pln = remove plane from each time slice

modelorder = (com,gai,ext,flt,ast,noi)
```

By default, the final values of these fitted models are *not* written to files. However, this can be modified by setting `exportndf` in the configuration file to the list of models that you wish to view:

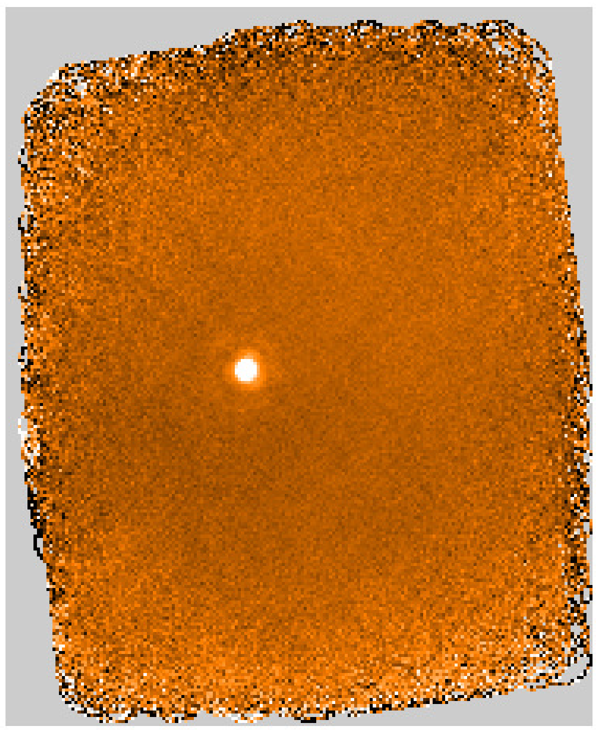

Figure 9: Map of Uranus produced with the SMURF task makemap using the iterative algorithm with default parameters. The S/N of the source is now greatly improved compared to the simplistic reduction in Fig. 7, and the negative ringing has been removed.

```
# Specify a value of 1 or 0 to export all or none of the components
# You can also specify an array of components to export using the same
# format as modelorder. Note that you can specify additional
# components 'res' and 'qua' to what may be provided to modelorder if
# you wish to export the residual model or quality arrays
# respectively. Exportation of 'res' is implied if 'noi' is specified
# as it becomes the variance components of the resulting NDF for
# 'res'. 'qua' will become the quality component of any full 3-dimensional
# model (e.g. 'res', 'ast', 'flt', 'ext'), but no quality will be
# written to model components with different dimensions.

exportndf = (com,gai,ast,flt,res,noi,qua)
```

If we make a local copy of `dimmconfig.lis`, add the above `exportndf` line, and re-run the iterative map-maker with this modified configuration, it now produces several new files at the end:

```
s4a20091214_00015_0002_con_ast.sdf
s4a20091214_00015_0002_con_com.sdf
s4a20091214_00015_0002_con_gai.sdf
s4a20091214_00015_0002_con_flt.sdf
s4a20091214_00015_0002_con_res.sdf
```

Note that the quality and variance for the data are stored as the VARIANCE and QUALITY components within the residual file NDF.

As with the input data, these are all standard Starlink NDF files which can be examined using all of the existing tools, and can be used by other SMURF routines such as sc2clean, sc2fft, and makemap. The base of the filenames is the first input file that went into the maps for each subarray, and the '`con`' suffix indicates that several data files may have been concatenated together. The new files are: '`*ast.sdf`', the time-domain projection of the astronomical signal as estimated in the final map (same dimensions as the input bolometer data); '`*com.sdf`', an estimate of the common-mode signal (predominantly sky emission and fridge temperature variations); '`*flt.sdf`', additional noise (low-frequency noise in this particular case) that was filtered out of each bolometer (same dimensions as the input bolometer data); and finally '`*res.sdf`', the residual bolometer signal once the other components have been subtracted from the original data, which should look predominantly like white noise (again, same dimensions as the input bolometer data).

Time traces for a single bolometer are compared for all of these model components in Fig. 10. Uranus is clearly seen as a positive spike in the astronomical signal component. The common-mode signal is the next largest, clearly exhibiting the 30 s fridge variations that are apparent in the raw data. The residual noise removed by the high-pass filter is significant, but much smaller than the common-mode component. Finally, the residual signal is quite flat, indicating that most of the signal has been accounted for in the other model components.

4.1 Configuration Files

Since the DIMM has a large number of parameters, several configuration files are supplied with SMURF for reducing common types of data. All of these files can be found in

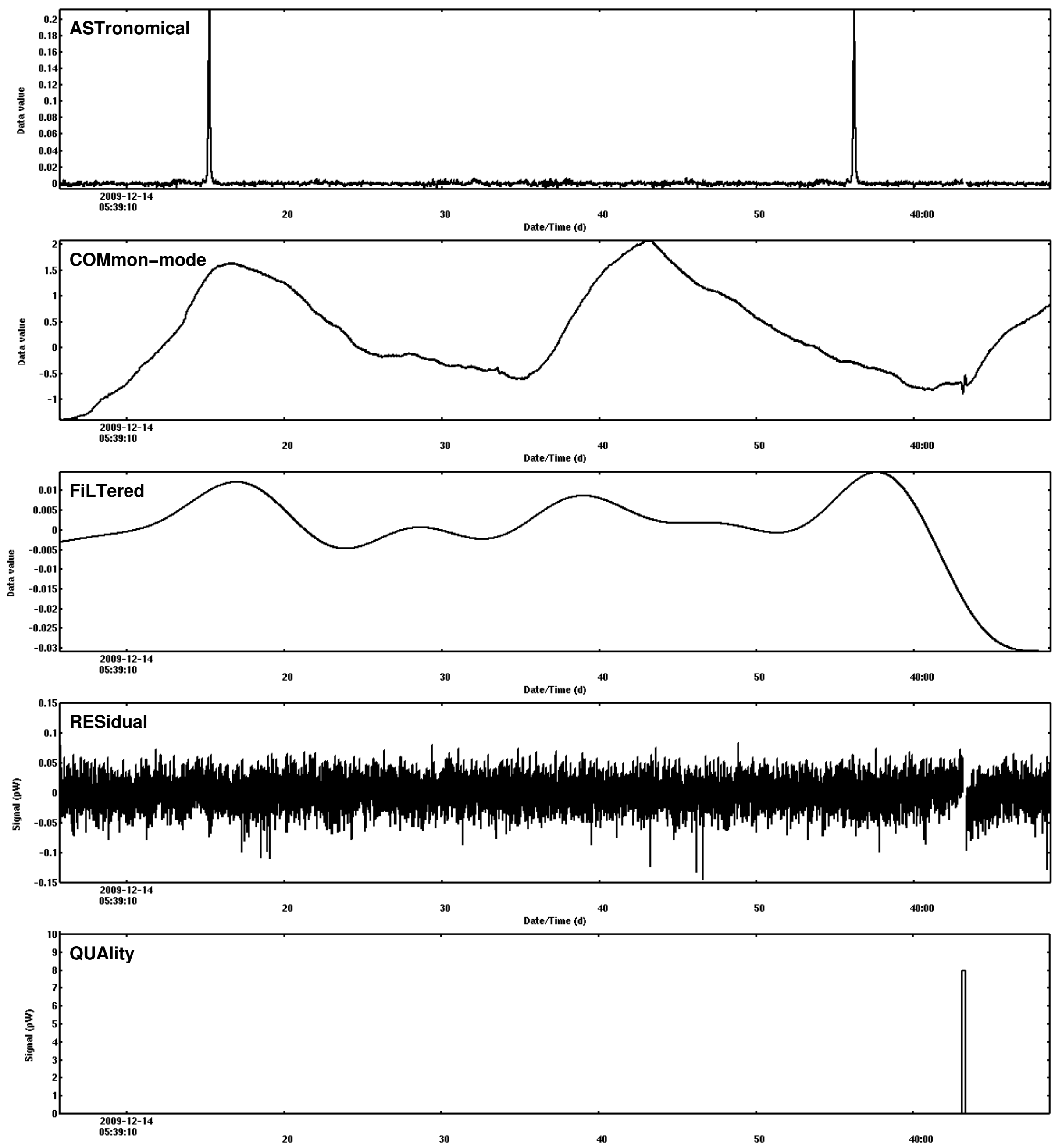

Figure 10: Time-domain components of the iterative solution for bolometer (13,23). From top to bottom: ASTronomical signal (clearly showing Uranus as positive spikes); COMmon mode (dominated by 30 s fridge variations); FiLTered (residual low-frequency noise missed by COM); RESidual (looking flat, except for a small residual around the location of a repaired DC step); and QUAlity (indicating the location of the DC step – the numerical value 8; this region of the data is not used in the final map). All of the plots were produced with GAIA, restricting the range of the x-axis to samples 2840–11500.

$STARLINK_DIR/share/smurf/ with filenames of the form dimmconfig*.lis. The default dimmconfig.lis should give reasonable results for most observations. Note that this file also contains the full set of parameters with extensive documentation. For clarity, all of the other configuration files for specific observations types are derived from dimmconfig.lis and simply override the relevant parameters.

While we have tested a wide variety of data sets with these files, it is certainly worth trying at least the default, and the specific configuration that sounds like it would be appropriate for your project:

dimmconfig.lis Before the iterative solution begins, some pre-processing steps are performed, like repairing DC steps, turning off particularly noisy bolometers (see the noiseclip parameter), and the mean levels are subtracted. This default configuration solves for the following models: COM, GAI, EXT, FLT, AST, NOI. The components COM and GAI work together to calculate the average signal template of all the bolometers, and then fit/remove the templates from each bolometer. They also identify outlier bolometers that do not resemble the template and remove them from the final solution. The component EXT applies the extinction correction (derived from the water-vapour radiometer by default), and then FLT applies a high-pass filter to the data, above frequencies that correspond to angular scales of 600 arcsec and 300 arcsec at 450 μm and 850 μm, respectively (as established automatically from the mean slew speed of the scan). These cutoff frequencies were chosen through trial and error, but appear to give good results under a wide variety of situations. Finally, AST estimates the astronomical signal contribution to the bolometer signals from the current map estimate, and NOI measures the noise in the residual signals for each bolometer to establish weights for the data as they are placed into the map in subsequent iterations.

dimmconfig_blank_field.lis This configuration is tuned for deep surveys for which the goal is to detect low-S/N point sources. Instead of iteratively applying a high-pass filter (FLT), which can result in convergence problems when there is little signal in the map, a single, harsher high-pass filter is applied as a pre-processing step (corresponding to 200 arcsec scales at both 450 μm and 850 μm). There are also more conservative cuts to remove noisy/problematic bolometers. Finally, the parameters ast.zero_lowhits and ast.zero_notlast are set to constrain much of the map to precisely zero for all but the last of 5 iterations. This helps with map convergence, and is a reasonable prior for a blank-field. Normally the map would then be processed using a matched filter (see Section 6.1).

dimmconfig_bright.lis This configuration is aimed at reducing maps of bright sources. Weaker noise clipping, DC step thresholds and common-mode rejection are used. In addition, the number of iterations is increased to a fixed value of 20 to allow potentially large-scale structures to converge.

dimmconfig_bright_compact.lis This configuration is aimed at reducing maps of bright, compact sources that are isolated at the centre of the map, and is derived from dimmconfig_bright.lis. The addition of ast.zero_circle and ast.zero_notlast parameters are used to constrain the map to zero beyond a radius of 1 arcmin for all but the final iteration. This strategy helps with map convergence significantly, and can provide good maps of bright sources, even in cases where scan patterns failed and the telescope degenerated into scanning back-and-forth along a single position angle on the sky.

`dimmconfig_bright_extended.lis` This configuration is for reducing maps of bright extended sources, and is also derived from `dimmconfig_bright.lis`. Here `ast.zero_snr` is used to constrain the map to zero wherever the S/N is lower than 5-σ. We recommend setting the parameter `itermap=1`, and then visually inspecting the maps produced after each iteration (e.g., `gaia map.more.smurf.itermaps`) to help determine an appropriate number of iterations.

If you want to modify parameters for a particular config file the best approach is to create a new file and first include a directive to load the file you want to modify and then supply your own parameters. You can see this technique is used in the installed configuration files to reference them all to `dimmconfig.lis`.

5 Calibrating SCUBA-2 Data

5.1 Extinction Correction

Analysis of the SCUBA-2 skydips and heater-tracking data from the S2SRO data has allowed calculation of the opacity factors for the SCUBA-2 450 µm and 850 µm filters to be determined. Full details of the analysis and on-sky calibration methods of SCUBA-2 can be found in Dempsey et al. (2010) [6].

Archibald et al. (2002) [7] describes how the Caltech Submillimeter Observatory (CSO) 225 GHz opacity, τ_{225}, relates to SCUBA opacity terms in each band, τ_{450} and τ_{850}. It was assumed for commissioning and S2SRO that the new SCUBA-2 filters are sufficiently similar to the wide-band SCUBA filters that these terms could be used for extinction correction. In the form $\tau_\lambda = a \times (\tau_{225} - b)$, the original SCUBA corrections were:

$$\tau_{450} = 26.2 \times (\tau_{225} - 0.014); \quad (1)$$

and

$$\tau_{850} = 4.02 \times (\tau_{225} - 0.001). \quad (2)$$

The JCMT water-vapour radiometer (WVM) is now calibrated to provide a higher-frequency opacity value which has been scaled to τ_{225}. The WVM (not the CSO 225 GHz tipper) data were used for this analysis.

The new filter opacities as determined from skydip data are as follows:

$$\tau_{450} = 19.04 \times (\tau_{225} - 0.018); \quad (3)$$

and

$$\tau_{850} = 5.36 \times (\tau_{225} - 0.006). \quad (4)$$

The SCUBA-2 filters are different from the SCUBA filters, with the 450 µm filter, in particular, significantly narrower than its SCUBA counterpart. The SCUBA-2 filter characteristics are described in detail on the JCMT website[6].

[6]http://www.eaobservatory.org/jcmt/instrumentation/continuum/scuba-2/filters/

The extinction correction parameters that scale from τ_{225} to the relevant filter have been added to the map-maker code. You can override these values by setting `ext.taurelation.filtname` in your map-maker config files to the three coefficients '(a,b,c)' that you want to use (where '`filtname`' is the name of the filter). The defaults are listed in `$SMURF_DIR/smurf_extinction.def`. We have also added a slight calibration tweak to WVM-derived values to correct them to the CSO scale. It is worth noting that if an individual science map and corresponding calibrator observation has already been reduced with the old factors (and your source and calibrator are at about the same airmass and if the tau did not change appreciably), any errors in extinction correction should cancel out in the calibration.

5.2 Flux conversion factors

Primary and secondary calibrator observations have been reduced using the specifically designed `dimmconfig_bright_compact.lis`. The maps produced from this are then analysed using tailor-made PICARD recipes. PICARD is a post-processing and data combination tool that uses the same infrastructure as ORAC-DR, but is designed to be used after the initial reduction with the DIMM is complete. Details of the PICARD recipes and how to use them can be found on the ORAC-DR web page[7].

A map reduced by the mapmaker has units of pW. To calibrate the data into units of janskys (Jy), a set of bright, point-source objects with well known flux densities are observed regularly to provide a flux conversion factor (FCF). The data (pW) can be multiplied by this FCF to obtain a calibrated map, and the FCF can also be used to assess the relative performance of the instrument from night to night. The noise equivalent flux density (NEFD) is a measure of the instrument sensitivity, and while not discussed here, is also produced by the PICARD recipe shown here. For calibration of primary and secondary calibrators, the FCFs and NEFDs have been calculated as follows:

(1) The PICARD recipe `SCUBA2_FCFNEFD` takes the reduced map, crops it, and runs background removal. Surface fitting parameters are changeable in the PICARD parameter file.

(2) It then runs the KAPPA beamfit task on the specified point source. The beamfit task will estimate the peak (uncalibrated) flux density and the FWHM. The integrated flux density within a given aperture (30 arcsec radius default) is calculated using PHOTOM autophotom. Flux densities for calibrators such as Uranus, Mars, CRL 618, CRL 2688 and HL Tau are already known to PICARD. To derive an FCF for other sources of known flux densities, the fluxes can be added to the parameter file with the source name (in upper case, spaces removed): `FLUX_450.MYSRC = 0.050` and `FLUX_850.MYSRC = 0.005` (where the values are in Jy), for example.

    An example of a PICARD parameter file (used for reduction of the 850 μm calibrators) is shown here:

```
[SCUBA2_FCFNEFD]
APERTURE_RADIUS=30.0
AUTOPHOTOM=1
MASK_SOURCE=1
BACKGROUND_FITMETHOD=fitsurface
FITSURFACE_FITTYPE=spline
```

[7] http://www.oracdr.org/oracdr/PICARD

```
FITSURFACE_FITPAR=4
USEFCF=1
FLUX_850.ARP220=0.688
FLUX_850.ALPHAORI=0.629
FLUX_850.TWHYDRAE=1.37
FLUX_850.V883ORI=1.34
LOGFILE=1
```

(3) It then uses the above procedure to calculate the three alternative FCF values described below.

- $\mathrm{FCF}_{\mathrm{arcsec}}$ (surface brightness calibration)

$$\mathrm{FCF}_{\mathrm{arcsec}} = \frac{S_{\mathrm{tot}}}{P_{\mathrm{int}} \times A_{\mathrm{pix}}}, \tag{5}$$

where S_{tot} is the total flux density of the calibrator, P_{int} is the integrated sum of the source in the map (in pW) and A_{pix} is the pixel area in arcsec^2, producing an FCF in Jy/arcsec^2/pW. This $\mathrm{FCF}_{\mathrm{arcsec}}$ is the number to multiply your map by when you wish to have surface brightness units, and to be able to carry out aperture photometry.

- $\mathrm{FCF}_{\mathrm{beam}}$ (point source calibration)

$$\mathrm{FCF}_{\mathrm{beam}} = \frac{S_{\mathrm{peak}}}{P_{\mathrm{peak}}} \tag{6}$$

producing an FCF in units of Jy/beam/pW.

The measured peak signal here is derived from the Gaussian fit of beamfit. The peak value is susceptible to pointing and focus errors, and we have found this number to be somewhat unreliable, particularly at 450 µm. $\mathrm{FCF}_{\mathrm{beam}}$ is the number to multiply your map by when you wish to measure absolute peak flux densities of discrete unresolved point sources. The peak value in the map is then the total flux density of the point source. You should not integrate over the source after calibrating in this fashion, as this will give an overestimate of the flux density.

- $\mathrm{FCF}_{\mathrm{beamequiv}}$ (improved point source calibration)

To overcome the problems encountered as a result of the peak errors, a third FCF method has been derived, where the $\mathrm{FCF}_{\mathrm{arcsec}}$ is modeled with a Gaussian beam, with a FWHM equivalent to that of the theoretical JCMT diffraction-limited beam at each wavelength. In effect, the resulting FCF gives an 'equivalent peak' FCF from the integrated value, assuming that the point source is a perfect Gaussian.

$$\mathrm{FCF}_{\mathrm{beamequiv}} = \frac{S_{\mathrm{tot}} \times 1.133 \times \mathrm{FWHM}_{\mathrm{beam}}{}^2}{P_{\mathrm{int}} \times A_{\mathrm{pix}}}, \tag{7}$$

or more conveniently:

$$\mathrm{FCF}_{\mathrm{beamequiv}} = \mathrm{FCF}_{\mathrm{arcsec}} \times 1.133 \times \mathrm{FWHM}_{\mathrm{beam}}{}^2, \tag{8}$$

where FWHM is 7.5 arcsec and 14.0 arcsec at 450 µm and 850 µm, respectively. This produces an FCF in units of Jy/beam/pW.

The $FCF_{beamequiv}$ and FCF_{beam} should agree with each other. However, this is often not the case when the source is distorted, for the reasons mentioned above. $FCF_{beamequiv}$ has been found to provide more consistent results and it is advisable to use this value when available, instead of FCF_{beam}. It is also advisable, when running the matched filter on data (see Section 6.1), to use the $FCF_{beamequiv}$ for calibration.

6 Examples of different reductions

6.1 Deep point source maps

Many deep SCUBA-2 observations are designed to detect unresolved point sources. In this example we work through the reduction of a cosmology survey, in which targets are high-redshift star-forming galaxies (although the procedure is applicable to any observation of faint, unresolved objects). Since the surface density of these distant sources falls rapidly with increasing brightness, most objects are, on average, only slightly brighter than the extra-galactic confusion limit – the flux density below which the surface density of sources is so great that there are many blended objects within a telescope beam. Consequently, the sources of interest are usually only a few standard deviations brighter than the noise in the map (caused by a combination of instrumental and source confusion). In light of this, the recommended strategy for reducing such maps involves two basic steps:

(1) Create a map using `dimmconfig_blank_field.lis` (see Section 4.1) which, compared to the default configuration, sacrifices structure on large scales to gain the best possible noise performance on small scales (i.e. making the map as flat as possible). This is a good compromise, since the sources should generally be the size of a telescope beam.

(2) Run the map through a combined 'matched filter' (which effectively fits a point spread function, or PSF, centered over every pixel in the map) and a background suppression filter (removing additional residual large-scale noise). This is a fairly standard technique used throughout the extra-galactic sub-millimetre community to identify potential sources.

For this example we will reduce a 13 minute, 450 µm, 6 × 6 arcmin CURVY_PONG map towards the galaxy cluster MS0451 (see Section 1 for instructions on obtaining the data for this tutorial and the usage policy). Although not deep enough to detect any individual sources, this example is useful for illustrating features that are common to most of the extra-galactic SRO data from the Spring of 2010.

First, assuming the data are in the current directory, we produce a map using the specialized `dimmconfig_blank_field`:

```
% makemap s4a20100313_00029_00\*.sdf map450 method=iterate \
config=^$STARLINK_DIR/share/smurf/dimmconfig_blank_field.lis
```

For comparison, we also make a map using the default configuration:

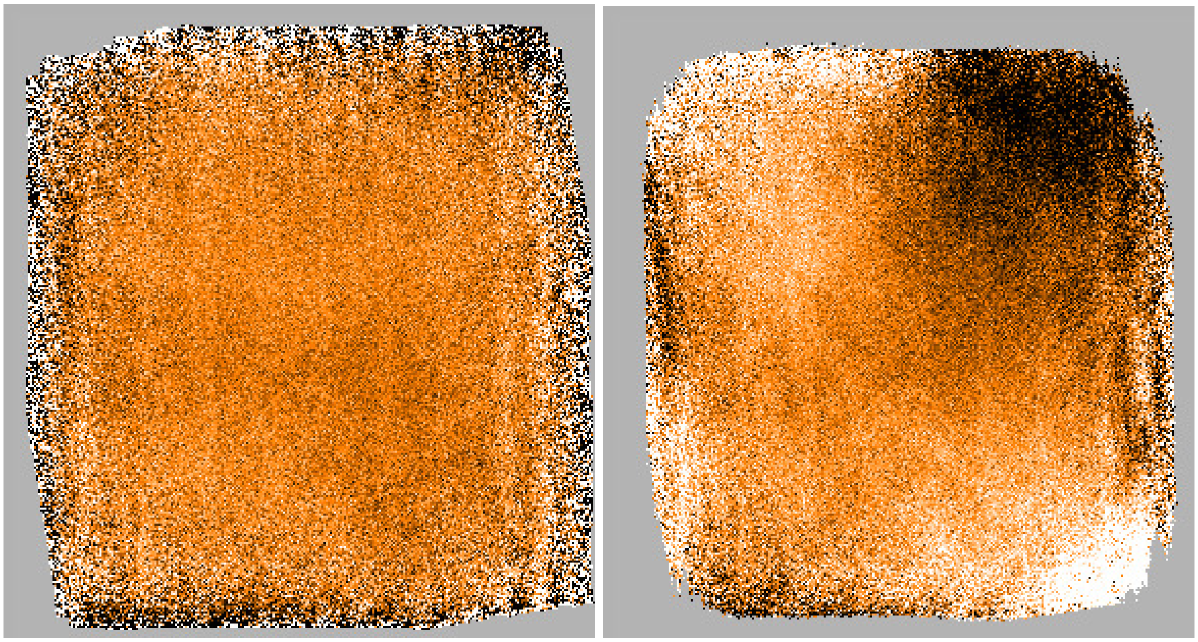

Figure 11: Maps of a deep cosmology field towards the cluster MS0451, at 450 μm. **Left:** map using the specialized `dimmconfig_blank_field.lis` which gives a significantly flatter result than **Right:** map using the default `dimmconfig.lis`.

```
% makemap s4a20100313_00029_00\*.sdf map450_default method=iterate \
config=^$STARLINK_DIR/share/smurf/dimmconfig.lis
```

Both of the maps are shown in Fig. 11. Clearly the specialized configuration yields a flatter map, although the white noise level is still quite large (no obvious sources are visible), and there is some residual structure in the map caused by low-frequency noise that is not effectively modeled/removed by the map-maker (vertical stripes that are aligned with the map edges).

In order to optimally find sources that are the size of the telescope beam, and suppress this residual large-scale noise, we provide a PICARD recipe (see Section 5.2) called `SCUBA2_MATCHED_FILTER`.

If there were no large-scale noise in the map, the filtered signal map would be calculated as follows:

$$\mathcal{M} = \frac{[M(x,y)/\sigma^2(x,y)] \otimes P(x,y)}{[1/\sigma^2(x,y)] \otimes [P^2(x,y)]}, \tag{9}$$

where $M(x,y)$ and $\sigma(x,y)$ are the signal and RMS noise maps produced by SMURF, and $P(x,y)$ is a map of the PSF. Here '$\otimes$' denotes the 2-dimensional cross-correlation operator. Similarly, the variance map would be calculated as

$$\mathcal{N}^2 = \frac{1}{[1/\sigma^2(x,y)] \otimes [P^2(x,y)]}. \tag{10}$$

This operation is equivalent to calculating the maximum-likelihood fit of the PSF centered over every pixel in the map, taking into account the noise. Presently P is simply modeled as an ideal Gaussian with a FWHM set to the diffraction limit of the telescope.

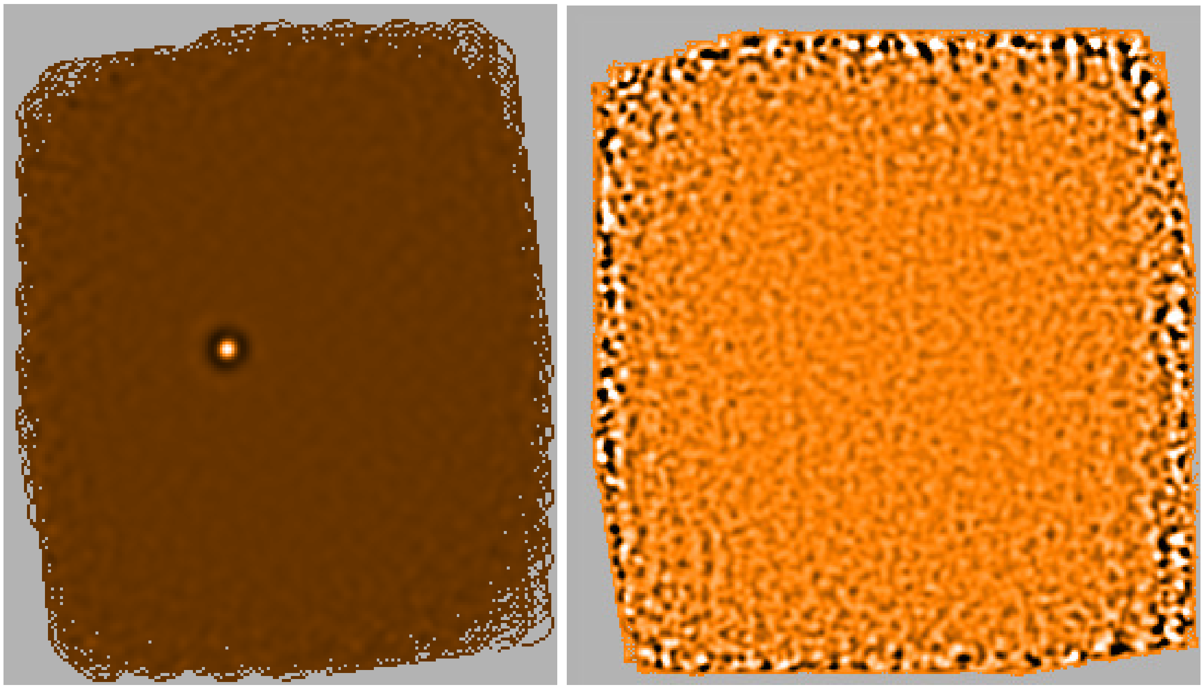

Figure 12: 450 µm maps processed with the PICARD recipe SCUBA2_MATCHED_FILTER, suppressing scales larger than 15 arcsec. **Left:** filtered map of Uranus from Fig. 9. **Right:** filtered version of deep cosmology map from left-hand panel of Fig. 11.

However, since there is large-scale (and therefore correlated from pixel to pixel) noise, the recipe also has an additional step. It first smooths the map by cross-correlating with a larger Gaussian kernel to estimate the background, and then subtracts it from the image. The same operation is also applied to the PSF to estimate the effective shape of a point-source in this background-subtracted map.

Before applying the filter to our cosmology data, we first look at the effect it has on the map of Uranus from Fig. 9. We create a simple parameter file called `smooth.ini`,

```
[SCUBA2_MATCHED_FILTER]
SMOOTH_FWHM = 15
```

where `SMOOTH_FWHM = 15` indicates that the background should be estimated by first smoothing the map and PSF with a 15 arcsec FWHM Gaussian. Next, the recipe is executed as follows:

```
% picard -recpars smooth.ini SCUBA2_MATCHED_FILTER uranus.sdf
```

The output of this operation is a smoothed image called `uranus_mf.sdf`. By default, the recipe automatically normalizes the output such that the peak flux densities of point sources are conserved. Note that the accuracy of this normalization depends on how closely the real PSF matches the 7.5 arcsec and 14 arcsec full-width at half-maximum (FWHM) Gaussian shapes assumed at 450 µm and 850 µm, respectively (an explicit PSF can also be supplied using the `PSF_MATCHFILTER` recipe parameter).

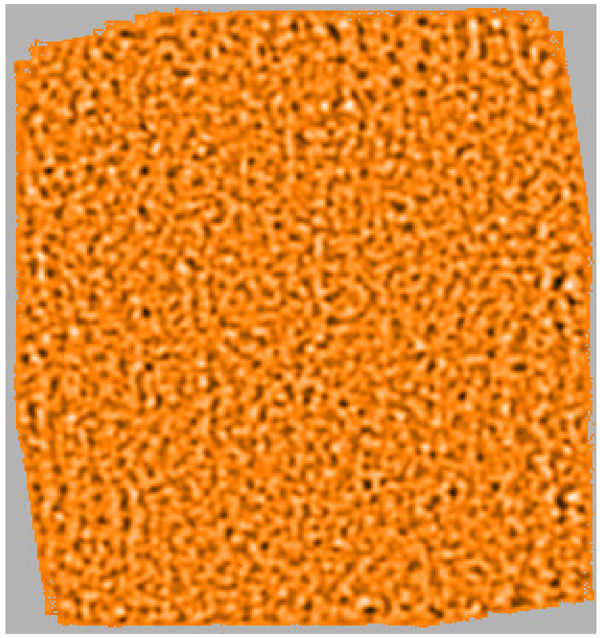

Figure 13: S/N map produced from the match-filtered image of the cluster MS0451 in Fig. 12, scaled from -4σ (black) to $+4\sigma$ (white).

The smoothed Uranus map is shown in Fig. 12. The map is generally flatter than the raw output of makemap, and the noise level is significantly reduced. However, the price that we pay for suppressing signal on scales larger than 15 arcsec is visible as the large negative ring around the source. For this particular case the dip is about 10 per cent of the peak signal. In addition to ringing, the filter attenuates the peak flux density of point sources. However, the normalization applies a positive correction to preserve peak flux densities, which results in an increased noise level.

Now that we know what this procedure does to a bright point source, we proceed to filter the map of MS0451:

```
% picard -recpars smooth.ini SCUBA2_MATCHED_FILTER map450.sdf
```

The smoothed map, `map450_mf.sdf`, is shown next to Uranus in Fig. 12. As hoped, this map has most of the remaining large-scale residual structure removed, and in general the noise is significantly reduced.

Finally, how should we find sources? The filtered map also contains a VARIANCE component, so it is easy to produce a S/N map using the KAPPA task makesnr:

```
% makesnr map450_mf map450_mf_snr
```

The resulting map, `map450_mf_snr`, is shown in Fig. 13. Compared to Fig. 12 the edges no longer appear as noisy because they have been down-weighted by the larger noise values where there were less data.

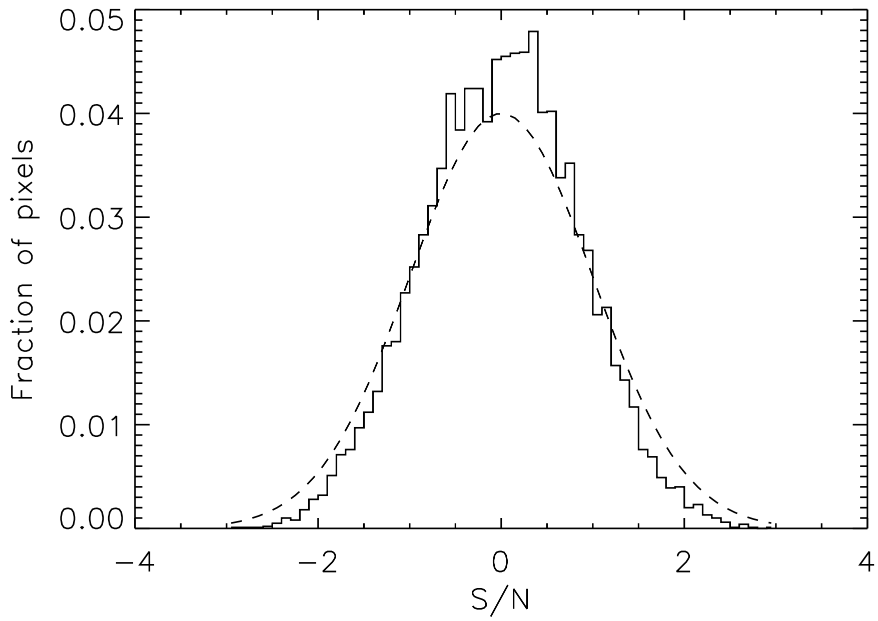

Figure 14: The distribution of S/N for the central 100×100 pixels of Fig. 12 (histogram), compared with an ideal Gaussian distribution with mean zero and $\sigma = 1$ (dashed line). The fact that the real distribution is narrower demonstrates that in this region of the map the noise is probably slightly over-estimated.

A basic procedure for identifying sources would be to locate peaks above some threshold S/N. However, as a word of caution, even after all of these steps the noise may not be perfectly well-behaved. In this example we do not expect any real astronomical source, so the S/N map *should* have a brightness distribution that resembles a Gaussian with standard deviation $\sigma = 1$ and mean zero. We perform this comparison for the central 100×100 pixels of the S/N map in Fig. 14, well away from any edge effects. In this case we find that the real distribution is slightly narrower than expected, suggesting that the noise has been mildly over-estimated.

We recognize that noise characterization is of utmost importance to the deep surveys, and we will continue to develop methods for estimating the true noise distributions in the final maps (e.g., using Monte Carlo simulations). Also, the Gaussian background noise suppression currently implemented in the matched-filter is isotropic. Clearly some of the residual large-scale noise has a preferred direction (such as the vertical stripes in Fig. 11). We are therefore investigating ways of automatically estimating more efficient filters for specific map geometries that will hopefully result in flatter maps, with reduced negative ringing around sources.

As a parting word on this subject, we mention some other tests that PIs should consider undertaking:

- Experiment with the size of the background suppression filter, as the large-scale noise depends on the scan pattern and state of the instrument when the data were taken. In this example, 15 arcsec was chosen in order to remove the bulk of the stripes parallel to the edges of the map. A smaller filter will cause more ringing, and more attenuation of the peak value (as mentioned above this is corrected for in terms of absolute calibration, but the S/N is reduced). On the other hand, a larger filter will leave more of the large-scale noise features.

- Split your data into mutually-exclusive subsets and produce independent maps. Are the highest S/N peaks detected in each of them?
- Use *jackknife* tests to verify the estimated noise levels, i.e. produce two maps from independent portions of the data and difference them (e.g., using the KAPPA task sub). This will remove any astronomical signal, but increase the noise by a factor of about $\sqrt{2}$. Is the standard deviation in the central pixels (where the noise should hopefully be uniform) roughly $\sqrt{2}$ larger than the noise estimated for either of the original maps?
- How many *negative* peaks above a given S/N are there compared to the *positive* peaks?

6.2 Extended Galactic Sources

In this section we shall focus on the reduction of extended, Galactic sources. In the following example we produce a map of the Orion nebula from two observations (#22 & #23) taken on 16th February 2010. Two alternative methods will be used. The first is more cumbersome, but illustrates the use of PICARD facilities for background removal, cropping, and mosaicking. The second method uses an alternative DIMM configuration aimed at maps of bright extended structures, and a single call to makemap.

6.2.1 Standard DIMM configuration + PICARD

If, as in this case, you have multiple observations contributing to your map, each observation can be reduced separately, and then combined to make the final map. Currently there is no advantage in terms of data quality to reducing all observations simultaneously or separately. However, the latter does allow the option of assessing the individual maps before coadding and is the method followed in this example.

When running the DIMM we select the default configuration file `dimmconfig.lis`; the individual parameters of which are described in Section 4.

```
% makemap '$STARLINK_DIR/share/smurf/s8d20100216_00022_000?.sdf' Orion22 \
method=iterate config=^$STARLINK_DIR/share/smurf/dimmconfig.lis

% makemap '$STARLINK_DIR/share/smurf/s8d20100216_00023_000?.sdf' Orion23 \
method=iterate config=^$STARLINK_DIR/share/smurf/dimmconfig.lis
```

The map from each observation is shown on the top row of Fig. 15. Note that these maps were observed as a series of rotating pong patterns to avoid repetition of any scan direction, hence their distinctive shape.

The default configuration file is designed to preserve maximum flux, however the difficulty of distinguishing between low frequency noise and real extended source emission inevitably means that some low frequency noise ends up in the final map. The maps show that although extended emission has been recovered the background is far from flat, displaying large scale patchiness as well as deep negative bowling surrounding the strongest sources.

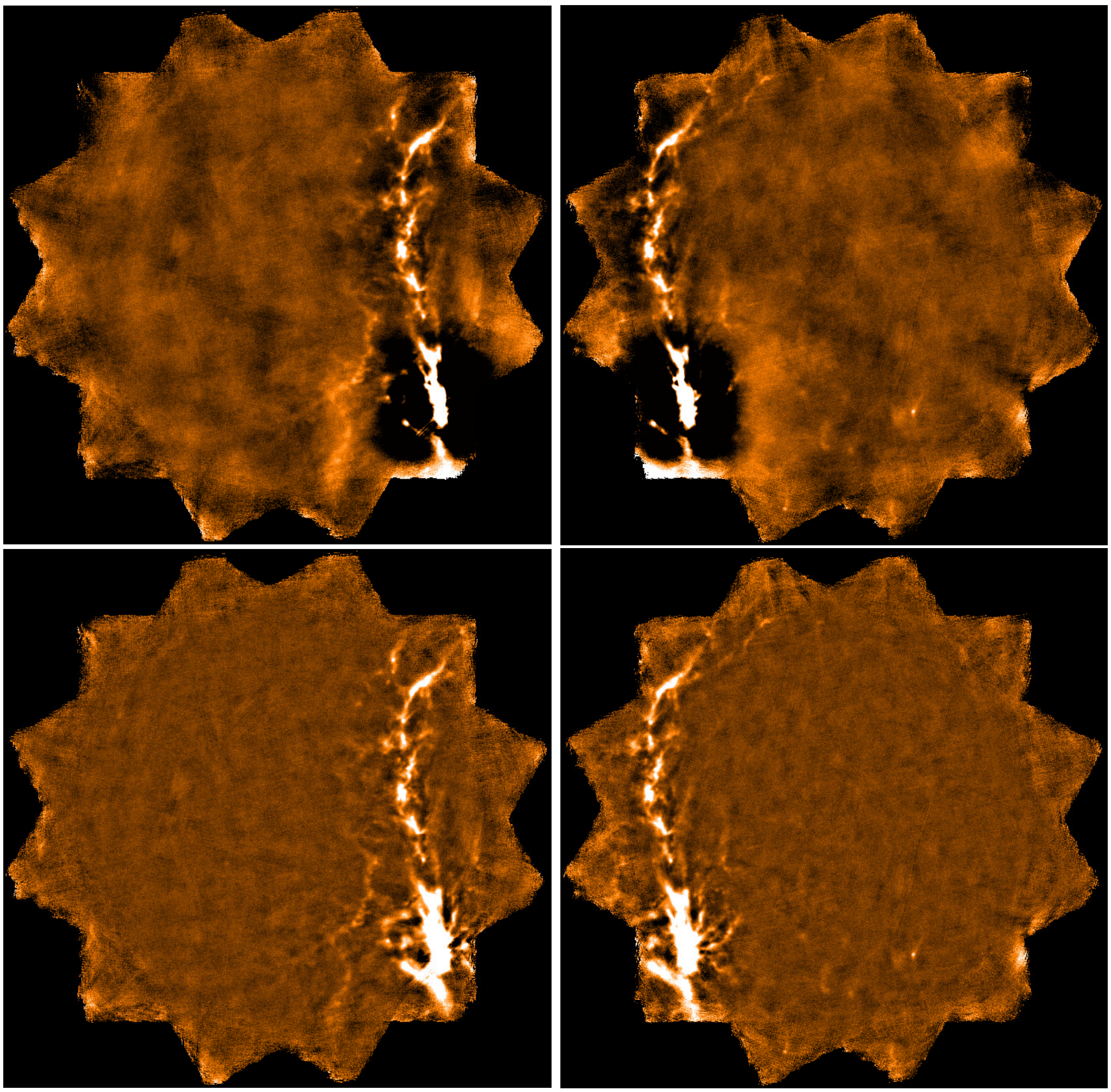

Figure 15: **Top row:** Orion22.sdf (left) and Orion23.sdf (right) maps from the mapmaker with no post processing. A patchy background and negative bowling around the strong sources is apparent. **Bottom row:** Orion22_back.sdf (left) and Orion23_back.sdf (right). The top row maps following processing with the PICARD recipe `REMOVE_BACKGROUND` using findback with a box size of 30 pixels (120 arcsec).

Both of these effects can be mitigated by post-processing using the PICARD recipe REMOVE_BACKGROUND. A number of different techniques are available in this recipe to control how the background is removed. In this example we have selected CUPID findback which uses spatial filtering to remove structure on a size scale less than that specifed by the parameter FINDBACK_BOX. The modified parameter file (params.ini) is shown below where the method is set to findback and the findback box size to 30 pixels or 120 arcsec when using 4 arcsec pixels:

```
[REMOVE_BACKGROUND]
BACKGROUND_FITMETHOD = findback
FINDBACK_BOX = 30
```

Caution is advised when selecting the box size, with a smaller box giving a flatter background but at the expense of source flux. This is of particular importance to extended sources where the recovery of faint emission is paramount.

```
% picard -recpars params.ini REMOVE_BACKGROUND Orion22.sdf
% picard -recpars params.ini REMOVE_BACKGROUND Orion23.sdf
```

The background subtracted maps are shown on the bottom row of Fig. 15 where both the bowling and uneven background have been significantly improved. Before we combine the maps we will first crop them to their originally requested size using the PICARD recipe CROP_JCMT_IMAGES.

```
% picard CROP_JCMT_IMAGES Orion22_back.sdf
% picard CROP_JCMT_IMAGES Orion23_back.sdf
```

These cropped maps are then coadded using MOSAIC_JCMT_IMAGES. This example utilises the default parameters where wcsmosaic with variance weighting is used for the mosaicking method although it can be configured to use makemos or to use a different wcsmosaic method.

```
% picard MOSAIC_JCMT_IMAGES Orion2*_back_crop.sdf
```

The key advantage to using the PICARD recipe over standalone KAPPA commands is that the exposure time image is also propagated correctly to the output mosaic (it is stored in the .MORE.SMURF.EXP_TIME extension).

The final, coadded map is shown in Fig. 16.

Note the output filename convention for each PICARD recipe: REMOVE_BACKGROUND creates output files with the suffix _back, CROP_JCMT_IMAGES creates files with the suffix _crop, while MOSAIC_JCMT_IMAGES creates files with the suffix _mos appended to the *last* input filename.

6.2.2 Bright / extended DIMM configuration

The simpler method is to make a map from all of the data at once using dimmconfig_bright_extended.lis:

```
% makemap "$STARLINK_DIR/share/smurf/s8d20100216_00022_0002[23].sdf" Orion \
method=iterate config=^$STARLINK_DIR/share/smurf/dimmconfig_bright_extended.lis
```

The large-scale noise and negative bowling is compensated for using a S/N-based zero mask during map-making. Everywhere the signal is below this threshold (5-σ by default), the map is set to zero for all but the final iteration. The resulting map, and the location of the iteratively-determined zero mask are shown in Fig. 17.

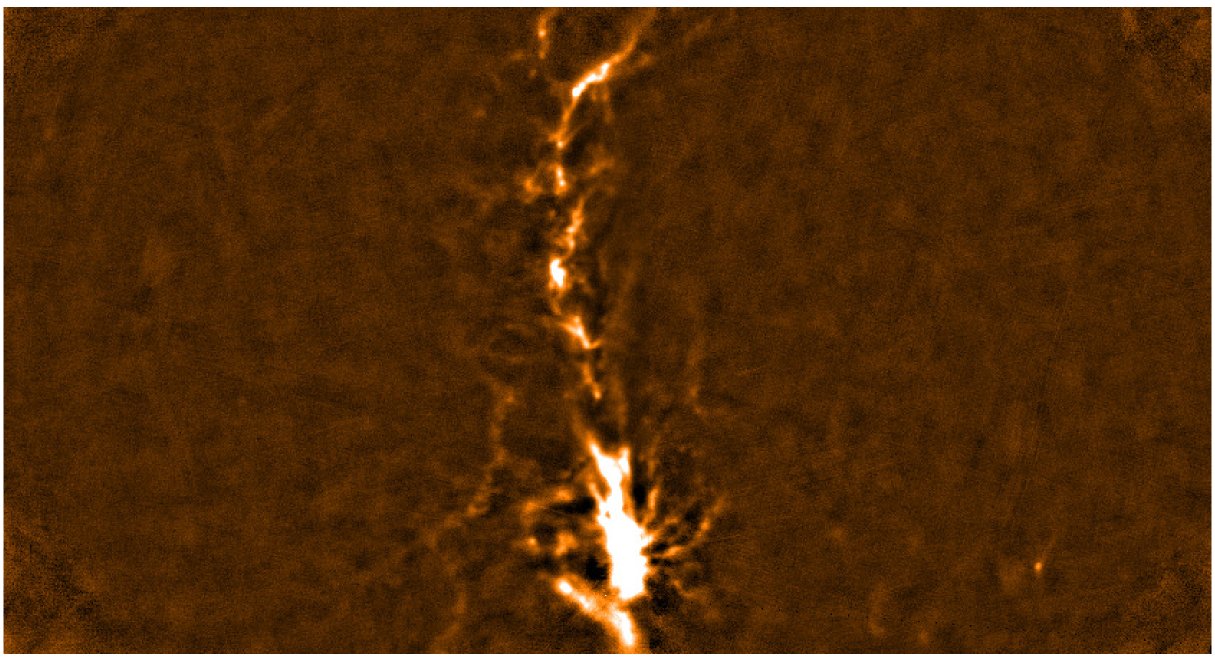

Figure 16: Orion23_back_crop_mos.sdf. The final map following the post processing steps of background removal, cropping and coadding with wcsmosaic.

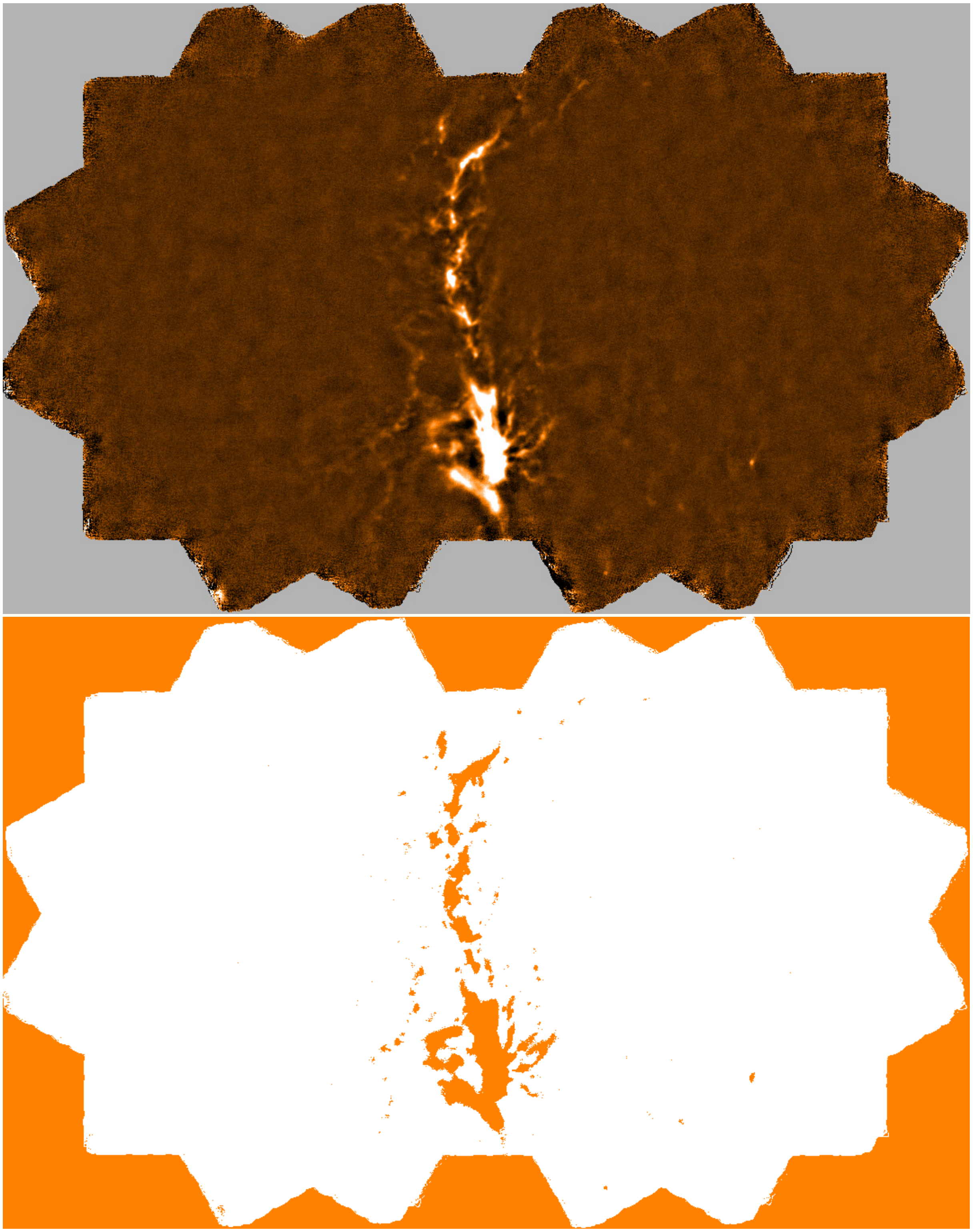

Figure 17: **Top:** Map using all of the same data as in Fig. 15, but processed simultaneously using a single call to makemap with the specialized `dimmconfig_bright_extended.lis`. **Bottom:** The QUALITY component of the resulting map indicates (white) where the map has been constrained to zero after all but the final iteration. It is selected based on a S/N threshold.

7 Cheat sheet for checking data quality

This section is a cheat sheet for a number of simple operations that can be used to do quick checks of the data quality.

7.1 Using sc2clean to help inspect time-series

sc2clean can be used to do two basic tasks in one go: concatenate data (with or without applying a flatfield); and cleaning (fix up steps and spikes, remove the means, filter, remove common-mode etc.). It uses the same configuration files as the iterative map-maker (though ignoring the map-making specific items).

In this first basic example, we just want to clean up some data enough to be able to see whether the bolometers have been flat-fielded correctly, and more-or-less exhibit the same behaviour over time:

```
sc2clean $FILES clean config=^$STARLINK_DIR/share/smurf/dimmconfig.lis
```

Here `$FILES` can just be a single file from a subarray, or a subset, e.g. `s8a20110417_00051_0003.sdf` (the first file containing science data), `s8a20110417_00051_000"[1234]"` (file 1 is a noise observation with shutter closed that gets ignored, file 2 is a flatfield observation that will be used to override the flatfield stored in the subsequent files 3 and 4 which are concatenated together, the `.sdf` is optional), `s8a20110417_00051_000\?` (files 1 through 9), `s8a20110417_00051_\*` (the whole observation). If you inspect the resulting `clean.sdf` in GAIA (Section 3.3) and flip through the data cube you should see all of the bolometers signals go up and down together with about the same amplitude: the hope is that for a well-behaved instrument you are mostly seeing sky noise variations that are seen with roughly the same amplitude by all bolometers. Another common feature, if the scans are particularly long and/or fast (e.g. 1 deg across), is strong periodic signals that are correlated with the scan pattern. See Section 3.2 – in particular you will want to plot `az` and `el` (the absolute azimuth and elevation), and also `daz` and `del` (the azimuth and elevation offsets from the map centre). This signal is usually azimuth-correlated due to magnetic field pickup. It only shows up in azimuth, because the instrument is on a Nasmyth platform and therefore does not move in elevation.

Part of the reason the signals look the same is because they have been flatfielded. You can turn off flatfielding using the `noflat` option to sc2clean, and you should then see that all of the detector amplitudes vary.

Another very useful option is to remove the common signal observed by all of the bolometers. This may be accomplished by

```
sc2clean $FILES clean \
   config='"^$STARLINK_DIR/share/smurf/dimmconfig.lis,compreprocess=1"'
```

The residual of this signal will exhibit second-order time-varying correlated signals across the focal plane. Usually these are not very large, but in some cases some very large localized signals have been detected, particularly in the 850 μm arrays in early 2011.

Another variation on this is to accentuate the residual low-frequency noise by low-pass filtering the result. This can again be accomplished by simply adding a filter command in the `config` parameter, which in this case low-pass filters with a cutoff at 10 Hz:

```
sc2clean $FILES clean \
   config='"^$STARLINK_DIR/share/smurf/dimmconfig.lis,compreprocess=1,filt_edgelow=10"'
```

Finally, in some cases you might just want to fit and remove polynomial baselines from the bolometers (by default only the mean is removed). This example will remove a line, but you can increase the value of `order` to remove higher-order polynomials

```
sc2clean $FILES clean \
   config='"^$STARLINK_DIR/share/smurf/dimmconfig.lis,order=1"'
```

7.2 Looking at calibration data

Currently all science observations are preceded by two calibrations: a short shutter-closed integration (file 1), followed by a "sky flat" to determine the response of all the bolometers to a current ramp through their heater resistors with the shutter open (file 2). The science observations are all subsequent files, except for the last which is also a sky flat.

The shutter-closed integrations can be useful to track the array sensitivity independent of sky conditions. You can run these files (or the science data files as well) through calcnoise (the NEP map for the subarray will be stored in the `.MORE.SMURF.NEP` extension of the output NDF `s8a_noise.sdf`):

```
calcnoise s8a20110417_00051_0001 s8a_noise method=! power=!
```

The sky flats can be used to calculate responsivity images using calcflat:

```
calcflat s8a20110417_00051_0002 s8a_flat method=! resp=s8a_resp respmask
```

The resulting `s8a_flat.sdf` contains a data cube of the data used to fit the responsivity (stacking several heater ramps together – it is easy to see bizarre/non-linear detectors that will subsequently be flagged and removed), and the actual responsivity image is stored in `s8a_resp.sdf`.

References

[1] Chapin E. L., et al., 2009, *SMURF – Sub-Millimetre User Reduction Facility*, Starlink User Note 258 1

[2] Currie M. J., Wallace P. T., Warren-Smith R. F., 1989, *Starlink Standard Data Structures*, Starlink General Paper 38.2 3.6

[3] Currie M. J., 1997, *KAPPA — Kernel Application Package*, Starlink User Note 95 1

[4] Draper P. W., 1997, *GAIA – Graphical Astronomy and Image Analysis Tool*, Starlink User Note 214 1

[5] Scott D., Van Engelen A., 2005, *Scan Mode Strategies for SCUBA-2*, SCUBA-2 Data Reduction document SC2/ANA/S210/005 1

[6] 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) 5.1

[7] Archibald, E. N., et al, 2002, *On the atmospheric limitations of ground-based submillimetre astronomy using array receivers*, MNRAS, 336, 1-13 5.1

[8] Jenness T., et al, 2002, *Towards the automated reduction and calibration of SCUBA data from the James Clerk Maxwell Telescope*, MNRAS, 336, 14-21 1

[9] Barnard, V. E., 2005,*A summary of the search for new Submillimetre Secondary Calibrators*, Tech. Rep. SCD/SN/011, Joint Astronomy Centre 1

A Data Characteristics and Issues

S2SRO data, due to our changing understanding of the instrument during shared risks observing, presents some particular challenges.

In order to illustrate in more detail some characteristics and issues associated with the SCUBA-2 data taken for the S2SRO, we will walk through an attempt to manually reduce a 850 µm observation on CRL 618 taken as observation 28 on 20100217.

A.1 Introduction data issues

There is no recommended procedure for manually reducing SCUBA-2 data, hence what follows is mainly intended to show the data issues rather than deliver a science product. Typically manual processing will remain inferior to the output of the iterative mapmaker. **Note:** a number of measures will be taken to address some of the issues seen during the S2SRO when upgrading SCUBA-2 to its full complement of 8 arrays. In particular in connection to the 30-sec fridge cycle, which dominates the common-mode signal, and a thermal gradient across the array that prevented the use of bolometers around the edge.

A listing of the observation directory shows the following:

```
 6956 s8d20100223_00017_0001.sdf
34140 s8d20100223_00017_0002.sdf
34140 s8d20100223_00017_0003.sdf
34140 s8d20100223_00017_0004.sdf
34140 s8d20100223_00017_0005.sdf
34140 s8d20100223_00017_0006.sdf
 6956 s8d20100223_00017_0007.sdf
```

As explained in the main section of the manual, the first and last files typically are 'dark' or 'flatfield ramp' observations. The remaining files can all be concatenated without resulting in too large a data set (which for science observations often will not be true unfortunately).

```
% cp (raw data dir)/*.sdf .
% sc2concat in=s8d\*.sdf out=sc17_con
```

A.2 Telescope tracking

A first inspection can be made of the tracking of the telescope by inspecting the JCMT state structure in the headers:

```
% jcmtstate2cat sc17_con > sc17_con.cat
% topcat -f tst sc17_con.cat
```

Select a 2-D image panel and the columns `x = TCS_TR_AC1, y = TCS_TR_AC2` to show the actual tracking on the sky. Each point marks the position of tracking center for each of the

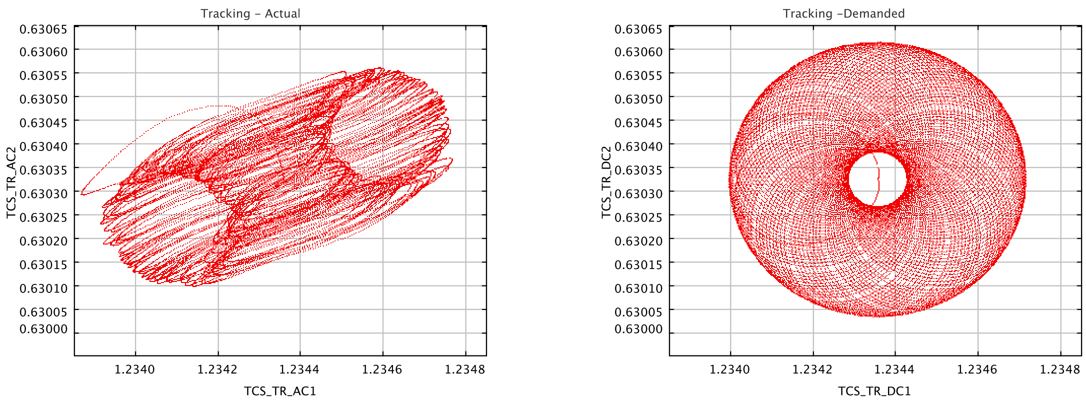

Figure 18: *left:* Actual tracking; *Right:* Demanded tracking

200 Hz samples. By contrast the demanded tracking can be plotted using `x = TCS_TR_DC1`, `y = TCS_TR_DC2`. The resulting figures are shown in Fig. 18.

Obviously, for this observation the telescope failed to follow the demanded daisy pattern due to its acceleration limits. This failure to follow the demanded daisy pattern is often seen at higher elevations (this observation was at an elevation of approximately 72 deg). The demanded daisy is 120 arcsec across, i.e. even for the failed pattern a 3 arcmin field was almost always covered by the footprint of the bolometer array.

Hence, although in general not disastrous for compact daisy fields, the distribution of the tracking points may result in more systematics across the field. In particular the diagonal pattern can result in a 'groove' in the final image when using the standard configuration file in DIMM. Switching to `dimmconfig_bright_compact.lis` can help.

A.3 Flat field

Note: As explained in §B.3.2, observations taken after 20100223UT include a fast flatfield ramp at their start and end. SMURF will use these to calculate the flatfield dynamically. Earlier observations have a flatfield in their headers calculated by the online system from an explicit flatfield observation taken some time prior to the observation. These flatfields are less reliable due to the variability of flatfields and the observations should be treated with caution. It is recommended that flatfield and copyflat should be used to re-calculate and re-insert the flatfield using the, now default, better ramp-fitting techniques.

The above sc2concat command applies the flatfield to the data (it does so by default). Collapsing the time series of the concatenated and flatfielded file to calculate the mean signal for each bolometer results in Fig. 19 showing the bolometer map for the s8d array used during the S2SRO: a number of *dead* columns can be seen as well as regions around the perimeter where the thermal gradient caused problems biasing bolometers.

The range of the mean value in the map is very large: from -28 to 30. An inspection of the cube shows that much of this is caused by differing DC levels across the bolometers. The DC term can be removed using sc2clean (mfittrend can be used as well). In general, for relatively compact

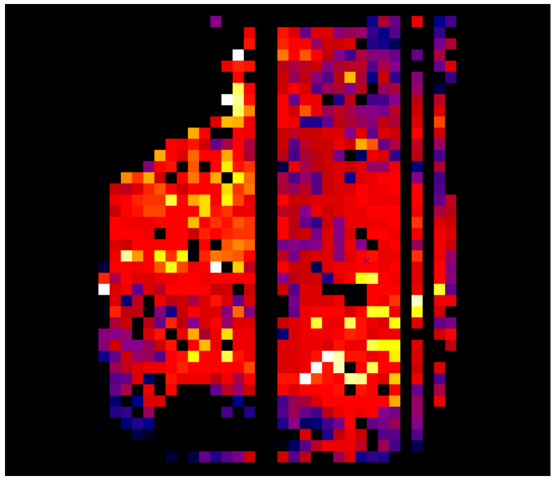

Figure 19: Mean signal in time-series

sources it should be safe to remove a first order baseline to also account for a monotonic drift component in the time series.

```
% cat my_sc2clean_bsl.def
order=1
dcfitbox = 0
spikethresh = 0

% sc2clean sc17_con sc17_conbsl config=^my_sc2clean_bsl.def
```

Collapsing the time-series cube again now results in a mean in a range of −3.0e-13 to 2.4e-13 as shown in Fig. 20. A histogram of the DC-removed data shows that the majority of the time-series data now are in a range of −0.1 to 0.1.

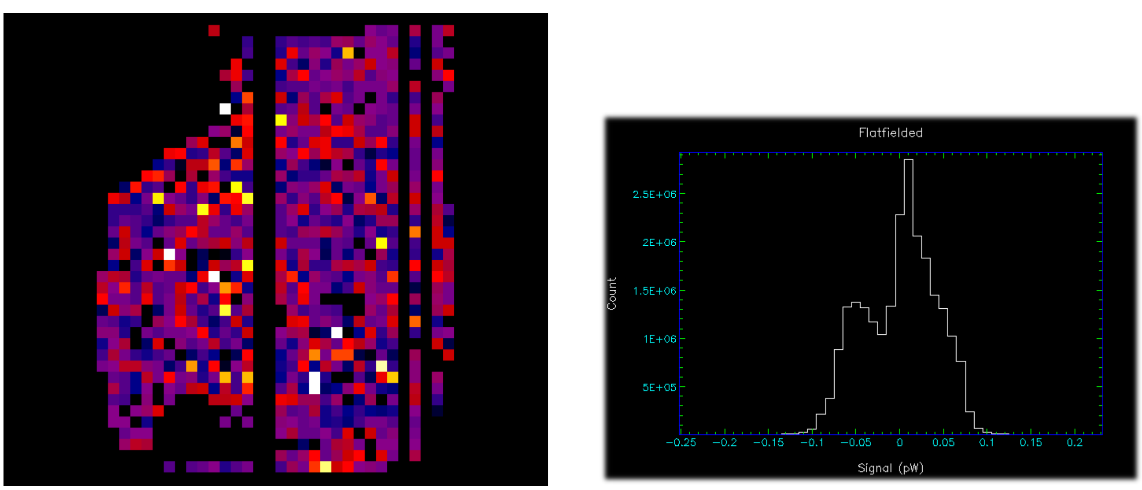

Figure 20: *left:* Mean signal in DC-removed time-series; *Right:* Histogram of values in DC-removed time series

A.4 Spikes and Steps

However, there are still significant outliers: the minimum and maximum pixel values are $\sim$8.1 and 3.4, respectively. Using GAIA to examine the time series in the DC-subtracted cube reveals remaining data issues as shown in the Fig. 21.

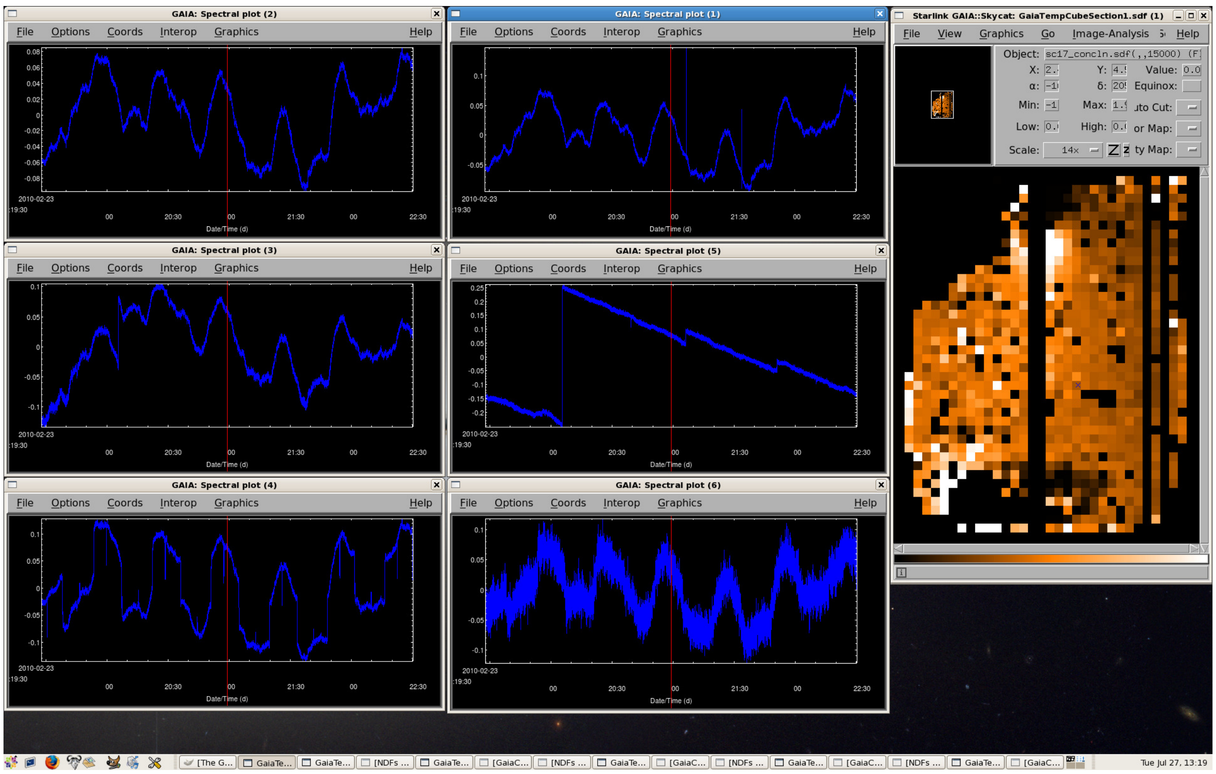

Figure 21: Example time series in the DC-corrected data set showing spikes, steps, and noisy bolometers. The dominant common-mode signal is a variation due to a 30 s temperature cycle in the dilution fridge. *Top-left:* a bolometer with a typical time series. Variations due to the 30 s fridge oscillation are obvious. Since all bolometers share this oscillation it is a 'common-mode' signal, but the amplitude may vary for different bolometers. *Top-right:* a bolometer with spikes in the time series. *Middle-left:* a bolometer with a moderate 'step'. Steps can be introduced by the bias for that bolometer 'rolling-over' as it reaches its limit. During the S2SRO the instrument software that flags these events was not active. *Middle-right:* a bolometer with a large step. *Bottom-left:* a bolometer with multiple steps and spikes. *Bottom-right:* a 'noisy' bolometer.

sc2clean is quite efficient in finding steps, but its spike removal is of limited effectiveness in the presence of a strong common-mode signal. As an example sc2clean was re-run with the following configuration file:[8]

```
% cat my_sc2clean.def
order = 1
dcfitbox = 30
```

[8](These parameters are explained in **SUN/258** (or run 'smurfhelp makemap config' or 'smurfhelp sc2clean config').

```
dcthresh = 25.0
fillgaps = 1
dcsmooth = 50
dclimcorr = 0
flagstat = 0
spikethresh = 5
spikebox = 50
noiseclip = 4.0

% sc2clean sc17_con sc17_concln config=^my_sc2clean.def
```

The results were that sc2clean left the time series of the top two plots in Fig. 21 unchanged, i.e. the common-mode variation was too large to cause the spikes in the right time series to be flagged. sc2clean completely flagged the two time series on the bottom row as bad. It did an excellent job correcting for the steps in the middle row time series, as shown in Fig. 22.

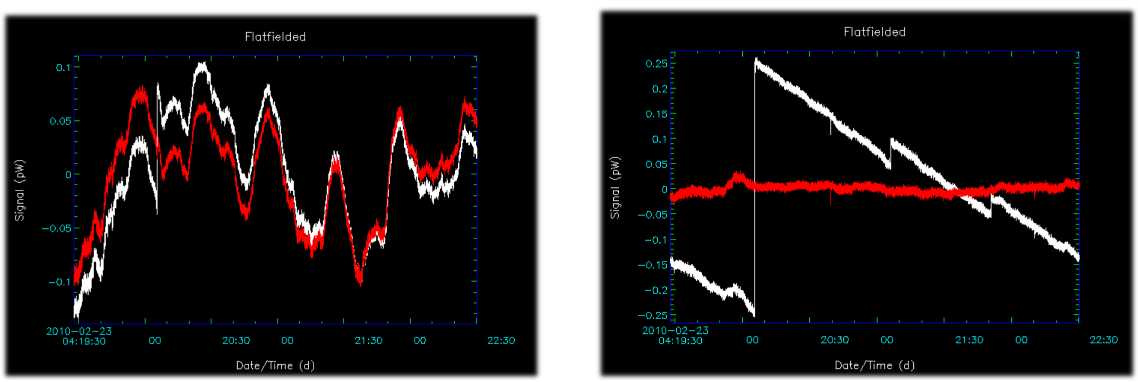

Figure 22: Original (white) and sc2clean step-corrected (red) time series.

A.5 Common Mode Signal

To investigate features of the time series further one needs to get rid of the dominant common-mode signal. Most of the variation seen is due to a 30-sec temperature cycle of the dilution fridge. This cycle affects the biasing of the bolometers and, in effect, varies the zero-level of each on that time-scale. There are various ways to remove this signal for quick inspection of the data, but here are two. Again, please be reminded that the aim here is not to reduce the data for map-making, but to illustrate characteristics of the data set.

Method 1: The simplest method is to mimic the action of the SMO module of the DIMM: subtract the median in a sliding box from the time series.

Method 2: Use the DIMM to subtract the common-mode signal and export the models for further inspection. The common-mode is captured by the `COM`, `GAI`, and `FLT` models as `GAI*COM+FLT`.

A.5.1 Method 1

The first method can be implemented rather simply using 'block' although the calculation of the sliding medians will take quite long. Since the time series corresponds to a path across the sky, this obviously suppresses any structures larger than the path corresponding to the box. However, it is a very efficient method to flatten the time series (including steps which can change into spikes) to allow an easy statistical analysis of the intrinsic noise characteristics of bolometers. This 'harsh selection' method may have merits for the analysis of point-source fields, but be aware that the DIMM in general will leave significantly more 'baseline' systematics in the time-series.

```
(csh)
set file = sc17_conbsl
echo "Subtract time-series block median"
block estimator=median in=${file} out=${file}_block \
      wlim=0 box='[1,1,200]'
sub ${file} ${file}_block ${file}_subblk
```

A block-size of 200 corresponds to 1 sec of data, which can be related to a spatial size through the maximum scanning speed e.g. 120 arcsec/sec. Remember though that the telescope spends a large fraction of the time at lower speeds either accelerating or decelerating. In fact, a block-size of 200 largely removed the signal from CRL 618 from this data set!

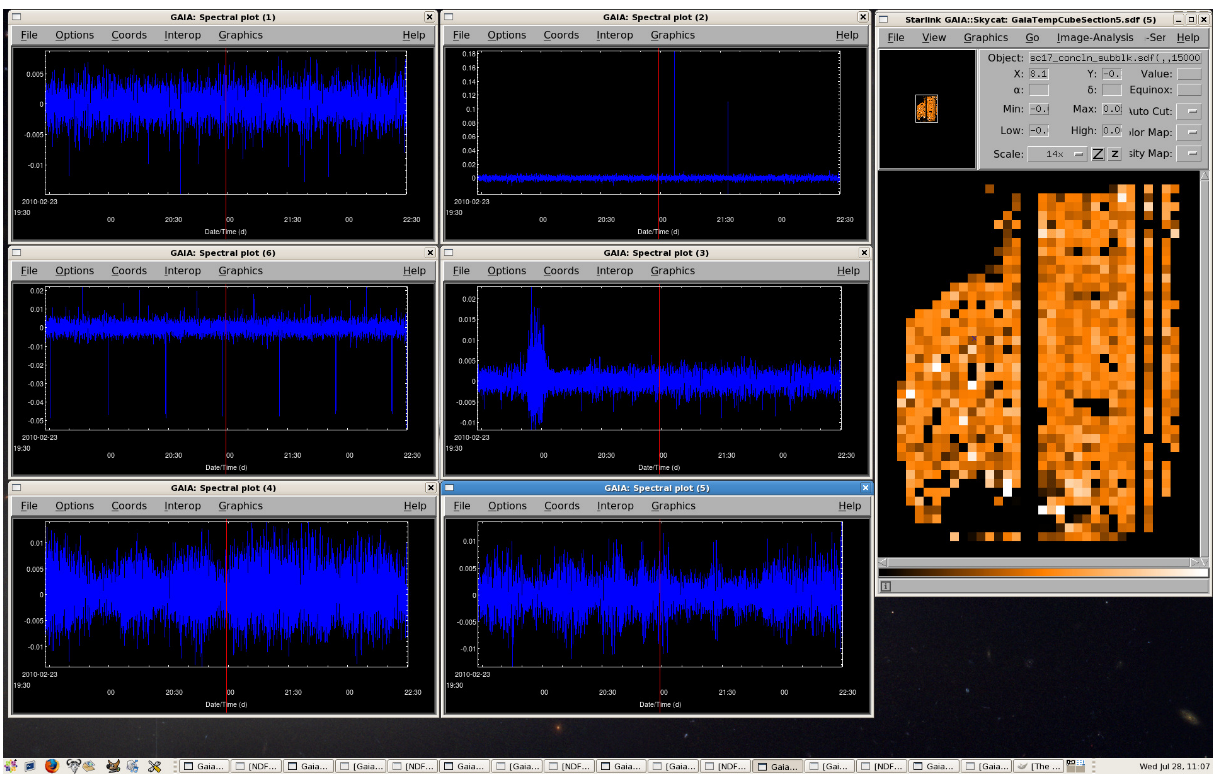

Figure 23: Example time series in the block-median common-mode subtracted data set highlighting intrinsic noise issues that can be seen in bolometers. The top-left time series is the same as the top-left time series in figure Fig. 21

An inspection of resulting file shows very flat time series. The top-left panel in the Fig. 23 shows a typical time-series after removal of a sliding median. The next two panel show time-series with negative and positive spikes. The middle-right panel shows a bolometer with an increased noise during part of the scan. The next two panels show bolometers with an uneven noise performance. Note that the DIMM will *not* specifically handle these issues, apart from de-spiking and an iterative clip of bolometers based on their overall noise level. It also is the case the residual variations remaining after a common-mode subtraction often are of a similar or larger level: the sliding-median method allows one to zoom in on the noise characteristics of individual bolometers and possibly derive a bad-bolometer map for use with the DIMM (many SCUBA-2 tasks accept a 'bbm').

A word of caution: at the high scanning speeds, point-like sources will look as spikes in the time series. At 120 arcsec/sec it takes the telescope at least 10 samples to cross the 45 μm beam. However, at 600 arcsec/sec (as may be used for large scan maps), the crossing happens in barely more than 2 samples!

One can attempt to further identify problematic bolometers by, for example, calculating the rms of each time series, but that falls outside the scope of this document:

```
collapse ${file}_block ${file}_block_rms estimator=rms \
        axis=3 variance=false wlim=0.0
```

A.5.2 Method 2

Method 2 is to use the DIMM to derive the common mode signal. The common-mode signal will be `GAI*COM+FLT`. There are a few gotchas to keep in mind though (note that some of these may change as the program gets further optimized):

- The DIMM will do a sc2clean in a preprocessing step which makes it hard to compare the models to the original file. At the least do an external DC removal first (as outlined above). Or do the sc2clean as an external step and switch off all sc2clean parameters in the dimmconfig file submitted to the DIMM.
- The DIMM will add padding on either side of the time series but not remove it from the models before writing them out. There currently is a bug that requires one to use 'setorigin' to properly align these cubes with the original one.
- The current `FLT` model will apodize (i.e. smoothly reduce the ends of the time series to zero; these ends are not used for the mapmaking). The apodization is needed for `FLT`'s current FFT, but a non-apodizing FFT method is under development.
- `com.notfirst` regulates whether the `COM` model is run during the first iteration. If set to 1, most of the dominant fridge-related variations will end up in the `FLT` model. A priori this is not a problem as far as removing the signal is concerned, but the `FLT` model works on each time series individually i.e. does not really derive a common-mode.
- Related to the previous point, if the fridge variations end up in the `FLT` model, the `GAI` model will not really show the gain variation across the bolometers, since the `COM` model in that case will only contain the residual common-mode variation (often dominated by ringing artefacts from `FLT`'s FFT). i.e. for this exercise we want to make sure that `com.notfirst=0`.

- The first plane of the GAI model has the multiplicative term but it is distributed around the median (or mean) gain rather than a value of 1 due to the way that the GAI and COM models have been implemented. i.e. divide the gain map by its median in order to get a value distributed around 1. However, note a subtlety: the gain of each bolometer in principle should have been fitted and accounted for by the flat field. Any gain differing from 1 in the GAI model can be interpreted as either a poor flat-field fit, the gain being variable, or some basic bolometers characteristics, e.g. their individual resistances, not (yet) being accounted for sufficiently.

For example, for these observations of CRL 618 modify the `dimmconfig_bright_compact.lis` configuration file as follows. Given that, at the time of writing, the FLT model apodizes, it has been left out.

*A reminder: this exercise aims at showing data features and **not** at showing how well the DIMM can handle these. For the latter one would want to run the DIMM with all its features enabled.*

```
^$STARLINK_DIR/share/smurf/dimmconfig_bright_compact.lis

numiter = 3                          # Just run a few iterations
modelorder = (com,gai,ast)           # Just do common mode part
exportndf = (com,gai,ast,res)        # Write models out
itermap = 1                          # Create map for each iteration
com.gain_box = 600000                # Single gain map for whole spectrum
order = 0                            # Allow for DC level adjustments
dclimcorr = 0                        # No correlated step detection/correction
com.notfirst=0                       # Make sure that COM is run before FLT
```

The above file exports all relevant models. It produces a moderately smoothed common mode time series and a single gain component for the whole observation. A script that handles combines the output models into a common-mode and common-mode subtracted cube is appended at the end of the document. It actually gives us three useful files to look at: the derived common-mode signal (`_commode`), the relative gains of the bolometers (`_gain`), as well as a common-mode subtracted cube (`_astres`).

The common-mode reduction script is appended at the end of this document.

Fig. 24 shows a typical time series with the fitted common-mode signal.

The input cube to makemap had 812 'good' bolometers, the derived gain map 651: makemap has flagged an additional 161 bolometers as bad. A quick inspection of the masked bolometers shows that the majority have steps, increased noise, or multiple spikes. The gain map itself ranges from 0.44 to 1.89 and a histogram shows that of the 651 unflagged bolometers 593 (∼90 per cent) are within a range of [0.75,1.25] and 622 within [0.65,1.35]. To some degree this range indicates that for the S2SRO data the flat field *in practice* was in general not very accurate or stable probably due to one or more of the aforementioned reasons.

For a further analysis one can also e.g. collapse the common-mode subtracted cube over the time-series to calculate the median and rms:

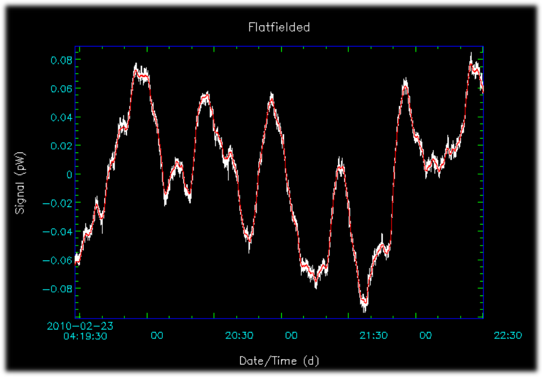

Figure 24: Example time series (white) and the derived common-mode signal (red). This time series is the same as the top-left time serie in Figs. 21 and 23

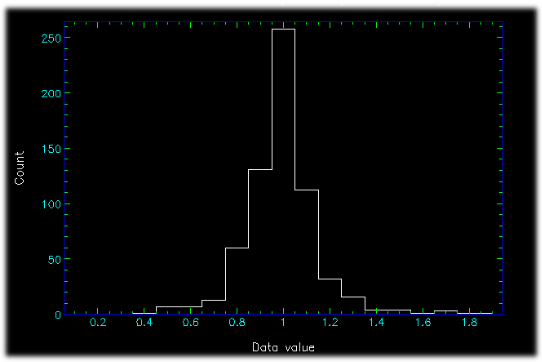

Figure 25: Histogram of the bolometer gains as derived by the DIMM based on their response to the common-mode signal. Note that these gains are, by default, only used for the subtraction of the common mode and not for the subsequent gridding into a map.

```
% collapse ${file}_astres ${file}_astres_median estimator=median \
           axis=3 variance=false wlim=0.0
% collapse ${file}_astres ${file}_astres_rms estimator=rms \
           axis=3 variance=false wlim=0.0
```

The median *signal* ranges from −33e-04 to 30e-04, with 582 bolometers falling within a range of −5e-04 to 5e-04. The median *rms* is 3e-03 with a maximum of 14e-03 and 578 bolometers below a rms of 6e-03 (twice the median). The three panels in Fig. 26 summarize this information.

```
% thresh ${file}_gain'(,,~1)' temp \
         thrlo=0.75 thrhi=1.25 newlo=0.0 newhi=2.0
% thresh temp ${file}_gainmsk \
         thrlo=1.5 thrhi=0.5 newlo=1.0 newhi=1.0
% thresh ${file}_astres_median'(,,~1)' temp \
         thrlo=-5e-04 thrhi=5e-04 newlo=-1.0 newhi=1.0
% thresh temp ${file}_astres_medianmsk \
         thrlo=5e-04 thrhi=-5e-04 newlo=0.0 newhi=0.0
% thresh ${file}_astres_rms'(,,~1)' temp \
```

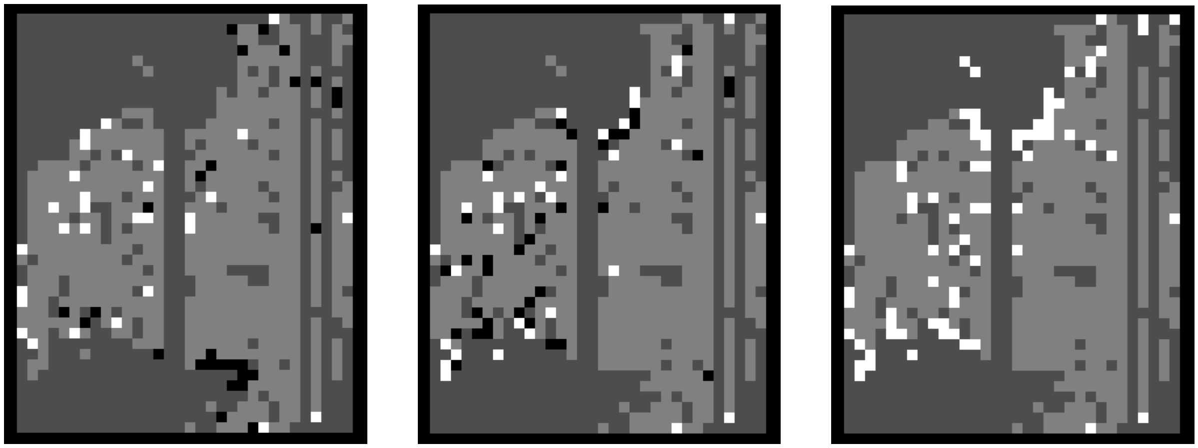

Figure 26: *Left:* Gain image: black < 0.75, white > 1.25; *Middle:* Common-mode subtracted median: black < −5e-04, white > 5e-04 *Right:* Common-mode subtracted rms: white > 6e-03 (twice the median).

```
                thrlo=0 thrhi=6e-03 newlo=-1.0 newhi=1.0
       % thresh temp ${file}_astres_rmsmsk \
                thrlo=6e-03 thrhi=0 newlo=0 newhi=0
```

The three maps have a significant subset of 'flagged' bolometers in common. An inspection of the common-mode subtracted data (`_astres`) shows that many of these bolometers have (multiple) steps that were not removed by sc2clean. Another subset shows variations that don't seem well modeled by the common-mode signal, although one has be careful not to mark the signature from CRL 618 as bad. But even for bolometers that pass through all the selection 'filters' there are quite a few that still have spikes, steps, or baseline ripples. Although the mapmaker was deliberately crippled for the above presentation, further development of the mapping algorithms will be needed to optimally handle SCUBA-2 data and produce the best possible maps.

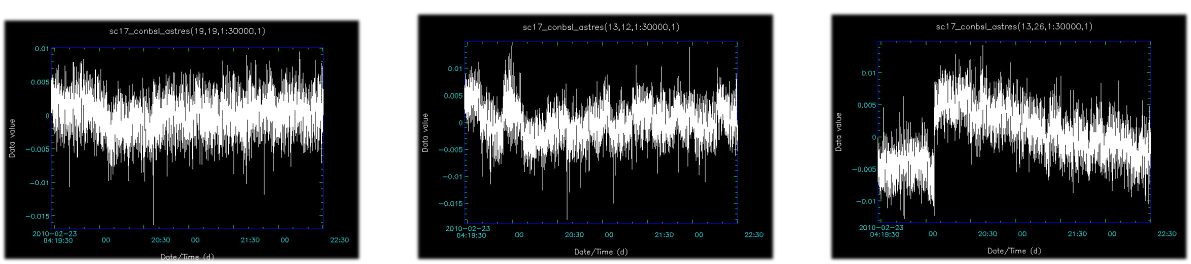

Figure 27: Sample time-series with a simple common-mode subtracted.

A.6 Maps

Although somewhat outside the scope of this document, after all this one might wonder what the maps from the various techniques looks like. Bear in mind that both a manual reduction as well as the mapmaker can be optimized better than is presented here. Apart from the first map, all the maps in Fig. 28 are presented with the same grey-scale stretch and show a 180 × 180 arcsec region around CRL 618.

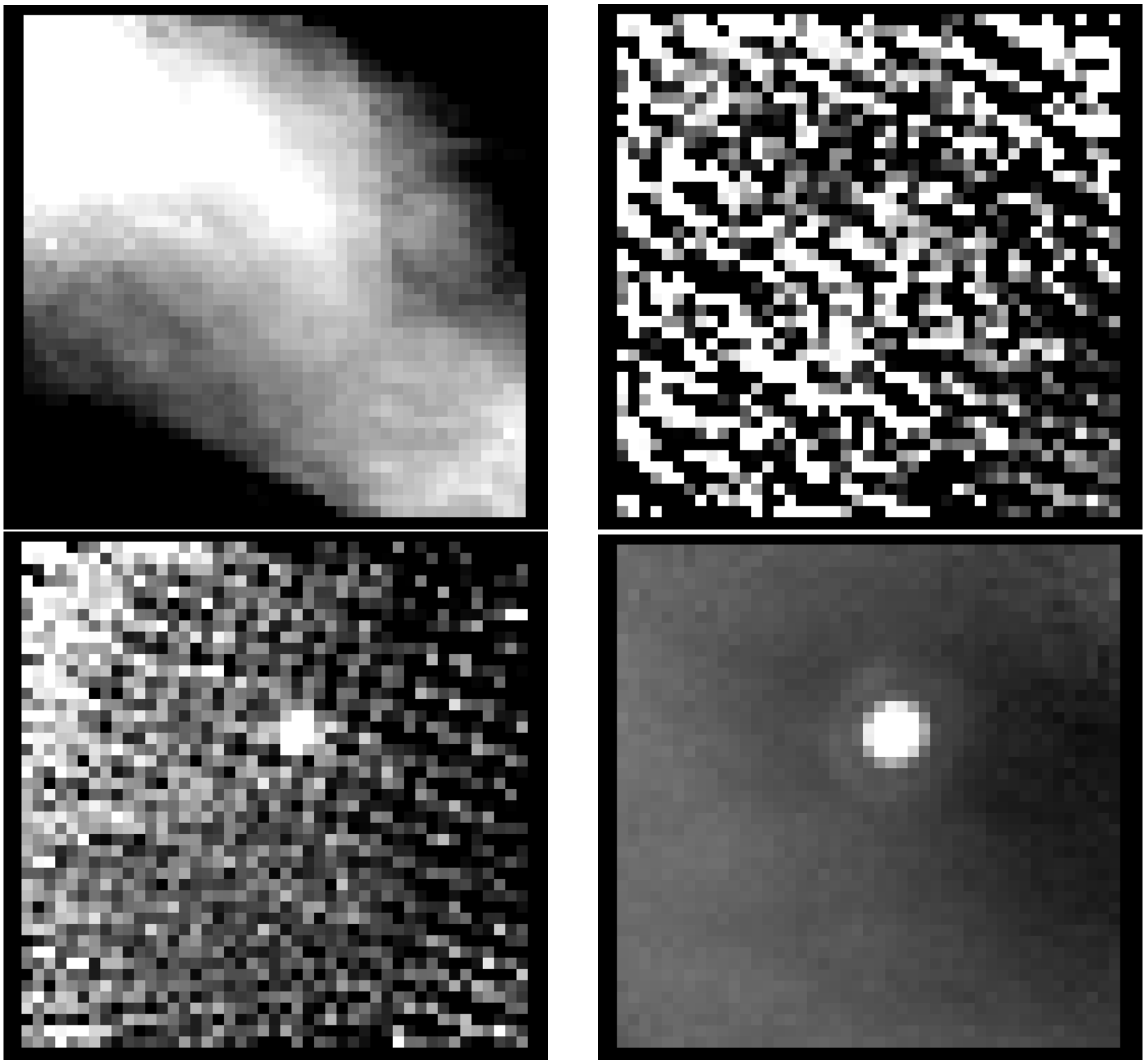

Figure 28: *Top left:* Map of the concatenated data; *Top right:* After removing a linear baseline from the time series; *Bottom left:* After linear baseline, steps, and spike removal; *Bottom right:* Iterative map maker with `dimmconfig_bright_compact.lis`

A.7 Sample Common-mode script

```
(csh)
echo "Remove linear baseline for easy comparison displays"
sc2clean sc17_con sc17_conbsl config=^my_sc2clean_bsl.def

set file = sc17_conbsl

echo "run the DIMM models"
makemap method=iter config=^my_dimmconfig_bright_compact.lis \
        in=${file} out=${file}_map

echo "# Get dimensions"
ndftrace ${file} >& /dev/null
set lbnds = `parget lbound ndftrace | head -1`
set ubnds = `parget ubound ndftrace | head -1`

set xlo =  $lbnds[1]
set xhi =  $ubnds[1]
set ylo =  $lbnds[2]
set yhi =  $ubnds[2]
set zlo =  $lbnds[3]
set zhi =  $ubnds[3]

echo "Find out padding in AST file
ndftrace ${file}_con_ast >& /dev/null
set lbnds = `parget lbound ndftrace | head -1`
set ubnds = `parget ubound ndftrace | head -1`

set zlo2 =  $lbnds[3]
set zhi2 =  $ubnds[3]
set pad = `calc "(${zhi2}-${zhi})/2"`
echo "$pad"
set zlo2 = `calc "${pad}+1"`
set zhi2 = `calc "${zhi}+${pad}"`

echo "Add unpadded part of ast and res model"
add ${file}_con_ast'(,,'${zlo2}':'${zhi2}')' \
    ${file}_con_res'(,,'${zlo2}':'${zhi2}')' ${file}_astres
setorigin ${file}_astres '[0,0,1]'
wcscopy ${file}_astres like=${file} ok=y

echo "Grow the COM back into a cube"
ndfcopy ${file}_con_com'(,,'${zlo2}':'${zhi2}')' \
        temp1 trim
manic  temp1  temp2  axes='[0,1]' \
       lbound=${xlo} ubound=${xhi}
manic temp2 temp_commode axes='[1,0,2]' \
       lbound=${ylo} ubound=${yhi}
setorigin temp_commode  '[0,0,1]'
wcscopy temp_commode like=${file} ok=y

echo "Grow the GAI back into a cube"
manic  ${file}_con_gai'(,,1)'  temp_gain \
```

```
       axes='[1,2,0]' lbound=${zlo} ubound=${zhi}
wcscopy temp_gain like=${file} ok=y

echo "Multiply GAI and COM to get the full common-mode cube"
mult temp_gain temp_commode ${file}_commode

echo "Derive 1-centered gain map"
histat ${file}_con_gai'(,,1)'
set median = `parget median histat | head -1`
cdiv  ${file}_con_gai'(,,1)' $median ${file}_gain
\rm temp?.sdf >& /dev/null
```

B Notes on Shared Risk Observing

B.1 S2SRO FCFs

Calibration observations were undertaken on a series of secondary calibrators, which are listed with their SCUBA fluxes in Table 1. The data for the FCF calculations were taken between the 23rd of February and the 14th of March, 2010.

Table 1: Secondary calibrators used for flux calibration of SCUBA-2. The flux values are sourced from the references noted in the table.

Source	RA(J2000)	DEC(J2000)	850 µm flux / Jy	450 µm flux / Jy	Ref
HL Tau	04 31 38.4	+18 13 59.0	2.36 ± 0.24	9.9 ± 2.0	[8]
CRL 618	04 42 53.60	+36 06 53.7	4.7 ± 0.37	12.1 ± 2.2	[8]
CRL 2688	21 02 18.81	+36 41 37.7	6.39 ± 0.51	30.9 ± 3.8	[8]
IRC+10216	09 47 57.38	+13 16 43.7	8.8 ± 1.1	17.5± 4.5	[8]
V883 Ori	05 38 19	-07 02 2.0	1.34 ± 0.01	7.28 ± 0.07	[9]
Alpha Ori	5 55 10.31	+07 24 25.4	0.628 ± 0.008	1.39 ± 0.04	[9]
TW Hydrae	11 01 51.91	-34 42 17.0	1.37 ± 0.01	3.9 ± 0.7	[9]
Arp 220	15 34 57.21	+23 30 09.5	0.668 ± 0.007	2.77 ± 0.06	[9]

The observations were reduced with the mapmaker using the config dimmconfig_bright_compact.lis and post-processed with the PICARD recipe `SCUBA2_FCFNEFD`. Figure 29 shows the 850 µm and 450 µm $FCF_{beamequiv}$ values for all calibrator observations taken during the S2SRO period. The resulting mean FCF's in each waveband are as follows:

$$FCF_{450} = 400 \pm 90 \text{ Jy/beam/pW} \quad (11)$$

$$FCF_{850} = 500 \pm 90 \text{ Jy/beam/pW} \quad (12)$$

The first note regarding the FCF's produced by the current reduction is in regard to the picowatt (pW) scale produced by the mapmaker. The pW scale is dependent on the accuracy of the heater resistance and the fraction of the heater power which is transferred to the bolometer. At the time of data release, the effective resistance was not well described, which induces an uncertainty in the pW scale of the resultant maps. This resistor value, and therefore the absolute pW scale, will be determined accurately when the instrument is returned to operation, and when this is known the new values will be added to the reduction code. This will affect the pW values of all observations, and the corresponding FCFs. However, the flux scaling will be preserved. The FCF values above have been calculated with the original pW scale.

Secondly, it is obvious that there is a large scatter in the FCFs in Figure 29. It is possible that the variation in the FCFs is produced by instrument performance changes or inconsistencies

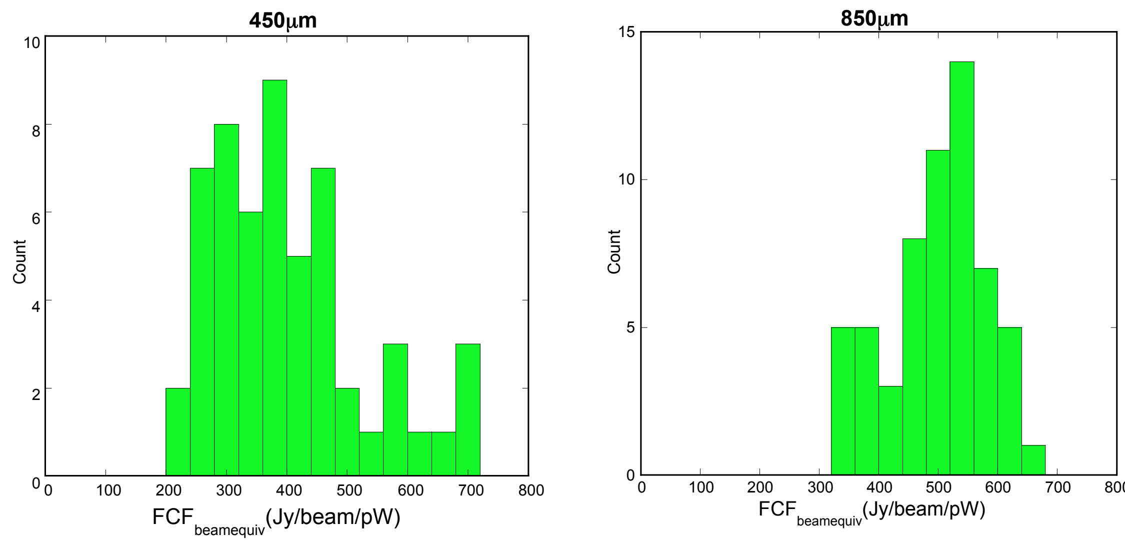

Figure 29: Histograms of the 450 μm (left) and 850 μm (right) $FCF_{beamequiv}$ values calculated for the secondary calibrators observed during the S2SRO period.

in the way the map-maker reduces the observations. At the time of release, the source of the scatter was not well understood and investigation continues. However, it is worth noting that the scatter between calibrations in an individual night of observations was sometimes observed to be as high as the scatter over the entire dataset. No trends were observed as a function of atmospheric transmission, or time during the night, or over the entire observing period. It is for this reason that it is advised that the average FCF values above are used, as opposed to selecting individual calibrations and using that FCF to calibrate your data.

B.2 Method for calibrating your data

It is the recommendation to use the mean $FCF_{beamequiv}$ values presented in the previous section to calibrate your data. However, if it is desired to produce an individual FCF from the night of a particular set of science observations, then the method is described here.

(1) Reduce the selected calibration observation using the `dimmconfig_bright_compact.lis config file`.

(2) Use PICARD's recipe `SCUBA2_FCFNEFD` on your reduced calibration observation. This will produce information to the screen and a logfile `log.fcfnefd` with the FCFs as mentioned above, and an NEFD for the observation. PICARD by default uses fixed FCF's to calculate the NEFD. (450 μm: 400 Jy/beam/pW and 850 μm: 500 Jy/beam/pW). If you wish to get an NEFD using the FCF calculated for the individual calibrator you are reducing, add `USEFCF=1` to your parameter file.

(3) Take your selected FCF and multiply your map by it using KAPPA cmult.

B.3 Significant Events

Part of the risk in S2SRO was that the instrument was being commissioned in parallel to being used for science observations. During February and March 2010 a number of events occurred that will possibly affect the data quality in a good or a bad way. This section documents these changes to aid in interpreting unexpected results that may come out of the data reduction process.

Note: all dates listed below are UT and are inclusive, and although setup changes should not affect the calibration they will affect the number and quality of functioning bolometers.

B.3.1 Data Files

Up to 20100219 the first and last file in a scan are dark frames for science, pointing[9] and focus observations. From 20100220 to 20100302 the first and last file are fast flatfield ramps. From 20100303 the first file is a dark frame and the second and last file are fast ramps. Note noise and discrete flat fields are different.

B.3.2 Flatfields

Beginning of S2SRO – 20100211

Until scan #17 on 20100211 discrete flatfields were reduced using the `TABLE` method.

20100211 – End of S2SRO

From scan #18 on 20100211 discrete flatfields were reduced using the `POLYNOMIAL` meth. *Note:* the stand alone flatfield observations are not used after the fast ramp flatfield was implemented 20100223. The fast flatfields are done as part of the observations.

20100213 – 20100215

Until scan #45 the heater step in the discrete flatfield was smaller leading to failure of the flatfield on the sky, particular at 450. Hence a lot of the flatfields on these dates were in the dark. This means less accurate flatfields.

20100220 – 20100222

Fast ramp flatfield used implemented but done in the dark. SMURF uses the discrete flatfields for these dates.

20100223 – 20100302

Fast ramp flatfield on the sky. *Note:* on 20100223 the data header claims the ramp was in the dark but from notes and the heater value it was on the sky i.e. the dark shutter header was wrong this night. SMURF knows about this and uses the fast ramp flatfield.

20100303 – end of S2SRO

Fast ramp flatfield used with an initial dark in the observations. This dark was used as a sanity check and is not used by the map-maker.

Explanatory Notes:

- Discrete flatfields step through a number of heater values rather slowly. The slow speed make the flatfield accuracy and quality susceptible to thermal drifts, 1/f noise and weather changes. Discrete flatfields were used for flatfielding until 20100223. Thereafter fast flatfield ramps are done at the start and end of every science observation.

[9]Pointings also have dark frames between the data frames in a scan for most of the SRO period.

- Fast ramp flatfield uses a fast heater ramp repeated several times - this improves the flatfield accuracy and quality.
- `TABLE` and `POLYNOMIAL` are two reduction methods used for discrete flatfields (fast ramp always uses a polynomial method). Due to the lack of dark frame subtraction the `TABLE` flatfield only used 2 points of the ∼10 heater settings in a discrete flatfield. Further, the slope from these two points were extrapolated far away. Obviously this affects the flatfield accuracy and quality. The flat field calculation should be redone using the `POLYNOMIAL` method for data taken before 20100211[10].

B.3.3 Setup

20100216 – 20100218

s4a detector bias set to 40000 - during the S2SRO the normal s4a value was 65000.

20100304 – end of S2SRO

The heater tracking was changed such that the heater is returned to the default value each time the shutter was closed. This was done to prevents drifts in the heater setting. Such drift affect the setup. However, noise observations also reset the heater. Thus the heater drift before this adjustment were small and is not believed to have affected the data.

B.3.4 Electronics

Beginning of S2SRO – 20100217

A large number of spikes are present in the s4a array data.

20100218 – end of S2SRO

The MCE was changed on the s4a array significantly decreasing the number of spikes in the s4a data.

B.3.5 Weather

20100310

Very bad seeing: data severely affected

20100311 early evening Bad seeing: data affected

B.3.6 Telescope

early commissioning – 20091202

For early commissioning data it was found that one of the mirrors was installed upside down leading to a slightly distorted beam shape. This was fixed from 20091203 so care should be taken when analysing very early commissioning data found in the archive before that date.

[10] Re-reduce the relevant flatfields using calcflat and then use copyflat to copy the flatfield into the data files before using the map-maker.

Beginning of S2SRO – 20100225

A two component pointing model (utilising only the `CA` and `IE` terms) was used, leading to large pointing shifts when doing large slews. On 20100226 the full eight component model was implemented and all-sky pointing improved. This should not affect data quality significantly since local pointing would still be adequate even with the earlier model.

C List of acronyms

CSO Caltech Submillimetre Observatory

DIMM Dynamic Iterative Map-Maker

FCF Flux Conversion Factor

FWHM Full-Width at Half-Maximum

GAIA Graphical Astronomy and Image Analysis Tool

JCMT James Clerk Maxwell Telescope

NEFD Noise Equivalent Flux Density

PSF Point Spread Function

S2SRO SCUBA-2 Shared Risk Observing which took place in February and March 2010.

SCUBA Submillimetre Common User Bolometer Array

SCUBA-2 Submillimetre Common User Bolometer Array-2

SMURF Sub-Millimetre User Reduction Facility

S/N Signal-to-Noise ratio

WVM Water Vapour radioMeter