SUN/236.5

Starlink Project
Starlink User Note 236.5

Paul Hirst
Brad Cavanagh
Joint Astronomy Centre, Hilo, Hawaii

October 2005

ORAC-DR – spectroscopy data reduction
5.0
User Guide

Abstract

ORAC-DR is a general-purpose automatic data-reduction pipeline environment. This document describes its use to reduce spectroscopy data collected at the United Kingdom Infrared Telescope (UKIRT) with the CGS4, UIST and Michelle instruments, at the Anglo-Australian Telescope (AAT) with the IRIS2 instrument, and from the Very Large Telescope with ISAAC. It outlines the algorithms used and how to make minor modifications of them, and how to correct for errors made at the telescope.

Contents

1 Introduction

ORAC-DR is a data-reduction pipeline operating at UKIRT, JCMT, and the AAT. It is part of the ORAC system. The pipeline reduces and displays multi-frame observations soon after they are read from the detector. This allows observers to assess the quality and suitability of their data in near real time. Yet ORAC-DR is capable of producing publication-quality results.

ORAC-DR is suitable for 'offline' data reduction at your home institution as well. There are many reasons why you may wish to use ORAC-DR in this fashion. For instance, you may have come back from UKIRT with only the raw observations; or there was an error in a telescope sequence (formerly an 'exec') mixing the groups of observations; or some data were reduced with a basic algorithm for speed at the telescope, and now you want to do a more careful job. ORAC-DR is capable of reducing data from instruments not running the pipeline at their respective telescopes. Hence ORAC-DR is available on Starlink.

SUN/230 presents an overview of ORAC-DR, general facilities like its display system, and it explains the differences between a pipeline and a traditional reduction package. Briefly, ORAC-DR uses a few data headers to direct the data reduction. Amongst these headers is the name of a *recipe*. A recipe is a series of high-level instructions such as "reduce, extract and flux-calibrate a spectrum" or "divide by a flat" that reduces an *observation* comprising one or more data frames. The implementation of each of these instructions is through a piece of Perl code—called a primitive—which calls Starlink packages such as CCDPACK and KAPPA, to actually do the processing of the bulk data.

This document describes how to use ORAC-DR software on Starlink to reduce data from the UKIRT spectroscopy instruments: CGS4 UIST, and Michelle; the AAT spectrometer IRIS2; and the ISAAC instrument on the Very Large Telescope (VLT). It outlines the various algorithms used in the recipes and includes detailed recipe documentation in the appendix. Besides the standard reduction recipes, this manual describes how you can customise recipes to suit your preferences and how to correct errors in the headers of your data frames.

There are complementary documents: SUN/232 describes the ORAC-DR for imaging from UFTI, UIST, IRCAM, Michelle, IRIS2, ISAAC and INGRID; SUN246 describes the ORAC-DR for integral field spectroscopy from UIST; and SUN/231 addresses the reduction of SCUBA data with ORAC-DR.

Those wishing to write their own recipes from scratch, or wanting to apply ORAC-DR to new instruments should consult SUN/233

2 Using the pipeline

2.1 Setting up ORAC-DR

Before you can run the pipeline you have to tell ORAC-DR for which instrument you wish to reduce data, the observation date, the directory containing the raw data, and where you want the processed data to be written. For the following two there are two options.

- The first needs your data to conform to the directory-naming convention of the instrument at UKIRT. This will be the case if you simply unpack the archive written by the **uktape** utility. In this case enter:

```
% setenv ORAC_DATA_ROOT <root_data_directory>
% oracdr_<instrument> <date>
```

  where <root_data_directory> is the directory in which you unpacked the data from the tape, <instrument> is either cgs4 or uist, and <date> is the UT date in the format YYYYMMDD. Note that each % represents the UNIX shell's prompt, which you do not type. The commands must be entered in the above order.

  For example, the standard location for raw CGS4 data is raw/ufti/YYYYMMDD/, and reduced/ufti/YYYYMMDD/ for the corresponding reduced data. If your data are stored in /home/users/abc/data/UKIRT/raw/cgs4/20031022/ you should enter the following:

```
% setenv ORAC_DATA_ROOT /home/users/abc/data/UKIRT/
% oracdr_cgs4 20031022
```

  to enable the pipeline for CGS4 data taken on 2003 October 22.

  Data taken from the AAT is handled differently, as there is no unified directory structure for either raw or reduced data directories. For IRIS2, INGRID or ISAAC data the best option is specifying where the raw and reduced data directories are, as shown below. ISAAC users should see

  Section 2.2.1 for a necessary preliminary naming conversion step.

- The second option is to separately define the raw and reduced data directories. Type the following:

```
% oracdr_<instrument> <date>
% setenv ORAC_DATA_IN <raw_data_directory>
% setenv ORAC_DATA_OUT <reduced_data_directory>
```

  The directories can either be given as full paths or as relative paths to the current working directory. Here is an example for UIST data using full paths:

```
% oracdr_uist 20040414
% setenv ORAC_DATA_IN /home/bradc/data/oracdr/asteroid/night1
% setenv ORAC_DATA_OUT /home/scratch/bradc/reduced
```

In the first case $ORAC_DATA_IN and $ORAC_DATA_OUT are still defined, but in terms of the root directory. For instance, re-using the earlier example with CGS4 for UT date 2003 October 22, $ORAC_DATA_IN points to $ORAC_DATA_ROOT/raw/cgs4/20031022/.

ORAC-DR operates in $ORAC_DATA_OUT, irrespective of what your current working directory is when you invoke it. Your current directory remains unchanged.

It is highly recommended to work in directories on disks local to the computer running the pipeline. Processing over NFS-mounted drives can be many times slower and can degrade the performance seen by other users. Running ORAC-DR on a Linux computer over NFS-mounted drives can also lead to corrupted files, crashing of the pipeline, or computer lockups.

2.2 Raw Data Formats and Conversions

Raw data take the form of multiple NDFs within an HDS container file for UKIRT data, or individual FITS files for AAT, INGRID and ISAAC data. For UISTthey comprise of one NDF for the data array and dynamic headers, such as the start time of the exposure, and another for static headers. For the HDS containers, initial reduction steps operate on each of the NDFs individually, only merging them to a simple NDF once the interleaving step is complete.

The Michelle HDS container also has NDFs for the individual chop beams. However, these cannot be merged until the data variance is calculated from the individual beams. Michelle reduced chopped data become simple NDFs once the recipe takes the difference of the two beams.

ORAC-DR automatically converts AAT FITS files into single NDFs in `$ORAC_DATA_OUT` which retain the original FITS headers. For INGRID, ORAC-DR converts a multi-extension FITS file into a multi-NDF HDS container file following UKIRT conventions.

2.2.1 ISAAC Preliminary Conversion

Since ORAC-DR as yet cannot cope with ISAAC file naming, which uses the UT epoch instead of a sequence number, there is a special C-shell script which must be invoked once, normally before the first ORAC-DR initialisation. If you enter

```
% isaac2oracdr
```

in a directory containing ISAAC FITS files, the command converts them into NDFs with names adhering to the UKIRT convention. The earliest file has observation number 1, and the observation number increments for each FITS file in time order. The script copes with file names in either the raw or archive nomenclature. It copes with data from more than one night in a given directory, assigning each night its own sequence of observation numbers; and it uses a common UT date for observations in a single night spanning midnight UT. You should put all of the calibration and target files for a given night in the same directory.

2.3 Running the pipeline

To run the pipeline, you use the **oracdr** command. This has a number of qualifiers described fully in SUN/230. There is online help too; enter

```
% oracdr -h
```

for a list of the options.

Unlike using ORAC-DR at UKIRT, you are unlikely to need the looping (`-loop` option) for offline processing, as all the data exist. Thus the most important qualifiers are `-list` and `-from`, which specify the frames to process; and the recipe name.

```
% oracdr -from 42
```

will process frames c20031022_00042 until the end of the night's data (assuming the earlier **oracdr_cgs4** command), running the recipes given by each frame's header (RECIPE keyword). More likely is that you provide a list of selected observations. The following example

```
% oracdr -list 41:49,51:59 POINT_SOURCE
```

processes frames from 41 to 49 inclusive and 51 to 59 inclusive, invoking the POINT_SOURCE recipe, and overriding the RECIPE header.

```
% oracdr -list 5,6,11,12
```

would reduce the frame 5, 6, 11 and 12. This is most likely to be applicable to pairs of flats and arcs.

There is a hazard with the `-list` option. Take care to select a complete set of frames associated with an observation. A common error is to accidentally include an arc frame not part of the group. Check the log on the raw data directory; it has file extension `.nightlog`. If you do not have a log, it is easy to create one.

```
% oracdr -from 1 -nodisplay NIGHT_LOG
```

This will create a log in `$ORAC_DATA_IN` for the current UT date. For CGS4, the log will be named `<date>.nightlog`. For multi-mode instruments such as Michelle, UIST, IRIS2, or ISAAC, there may be two log files created, one called `<date>_im.nightlog` and another called `<date>_sp.nightlog`, depending on the observing mode. In general mode-agnostic observations such as array tests are taken under imaging mode and will show up in the `_im` log, whereas science and calibration observations will show up in the `_sp` log.

2.4 Graphical initialisation and operation

You may prefer the ORAC-DR graphical interface called **xoracdr**. (See SUN/230.) It allows you to configure ORAC-DR: set the instrument, UT date, raw and reduced directories; and to run the pipeline with the various options. It permits monitoring of the primitives during execution of a recipe. **xoracdr** offers access to other facilities like display control and recipe editing. The in-built documentation does not pertain to the GUI itself but to general ORAC-DR information, however, **xoracdr** is straightforward to use and explore. While **xoracdr** has some rough edges, it is popular with many users. To try it, enter

```
% xoracdr &
```

Once the tool appears, you should select an instrument from the menu on the left, a UT date in the top centre, and raw and reduced directories to the lower right. The `From:` and `To:` refer to the observation numbers to process. When you are ready to reduce data, click on the `Start ORAC-DR` button.

2.5 Display

ORAC-DR optionally lets you inspect the raw frames, and the processed data as they are created. There is a variety of graphical methods available, including histograms and contour plots, if you choose a KAPPA GWM widget. Most people prefer a simple scaled image display with GAIA. This offers facilities to inspect and analyse the data, and both pixel and sky co-ordinates of the cursor position are presented. The selection of frame types to display, where they should appear, and how they are scaled are configurable using a simple text file or a special GUI tool **oracdisp**. See SUN/230 for details and examples.

Processing offline, there is less need to see the data displayed in real time. If you wish to accelerate the processing switch off the display option.

```
% oracdr -nodisplay ...
```

2.6 Calibration Information

ORAC-DR records calibration information, such as arc frames, flat fields, and the read noise, within index files, one for each type of calibration information. When the pipeline needs a calibration frame it searches the index file for the best matching entry subject to a set of rules. Each recipe reports the calibrations it has used. If no suitable calibration exists, the pipeline exits with an error message stating this fact. For further details see SUN/230.

Section **??** has an example of an index file.

You can also select a specific calibration using the `-calib` command-line option, provided the chosen calibration has an entry in the appropriate index file. See the section on calibration options in SUN/230 for details and examples.

2.6.1 Available calibration methods

The following calibration methods are available for spectroscopy recipes.

- bias — Use the given bias frame.
- flat — Use the given flat frame.
- mask — Use the given bad-pixel mask.
- readnoise — Use the given value for the detector readnoise in electrons.

2.7 Log files

In addition to presenting the progressing data reduction to an ORAC-DR X-window, ORAC-DR, by default, retains a copy of the processing steps and errors in a log file. These logs are important if something has gone wrong, and you have exited the X-window. Information from the applications software can be included if you run the pipeline with the `-verbose` command-line option. Logs also serve as a record of the data processing. Yet the log files are often overlooked because they are hidden. The log file is called `$ORAC_DATA_OUT/.oracdr_<number>`, where `<number>` is the current process identification. The `-log f` option to the **oracdr** command enables log-file creation.

See SUN/230 for details of the logging options.

3 Features of the Primitives

Primitives are the Perl scripts which actually call the applications to do most of the data processing. All of the spectroscopy recipes are, in principle, independent of the instrument. however, some recipes are inappropriate; for example, the LAMP_FLAT recipe is intended for flat-field generation for NIRI, but not for other instruments.

Not all the following steps apply to all recipes. Consult the reference section to see summaries for each recipe. The steps are presented in normal order of appearance.

The main primitives pertinent to each step are listed in bracketed italics, should you wish to tailor the recipes. These are found in the `$ORAC_DIR/spectroscopy` tree, unless they start with *general/*. Note that some may be instrument-specific variants, either given explicitly or with the *<instrument>* token, which means substitute the instrument name in uppercase.

While the following listed primitives form the bulk of the primitive library, there are many not listed here, mostly those for

recipe initialisation called <recipe>_HELLO_ (see Section **??** for more information), and for

recipe steering which control when to perform certain operations, called <recipe>_STEER_ or <recipe>_CONFIG_ (see Section **??**). The first of these is normally left unchanged unless there is a need to add more steering parameters. Other primitives not mentioned here are tied closely with single recipes, usually to create and file calibrations.

3.1 Preparation of Single Frames

[_REDUCE_SINGLE_FRAME_, <instrument>/_INSTRUMENT_HELLO_, _SPECTROSCOPY_HELLO_]

3.1.1 Manipulation of Raw Data

The first step copies the raw data into `$ORAC_DATA_OUT`. For instruments whose raw data is in FITS format, this step converts the raw data into NDF. For HDS container files, these are copied over into new HDS container files.
[_MAKE_RAW_FILE_]

3.1.2 Preliminaries

There are a few operations applied to all frames. First, history recording is switched on. It is recommended to leave this enabled, since it provides a record of the processing steps of your final spectra. Otherwise the pipeline becomes something of a black box. use the KAPPA command **hislist** to list the history records.
[_TURN_ON_HISTORY_]

The next step is to set the origin of the frame so that frame pixels retain the detector pixel indices. It then becomes possible to use a full-sized bad-pixel mask or a flat field on a subset of a detector's pixel grid.
[_SET_ORIGIN_]

For CGS4 data taken before 2000 August 13, the slit angle in the SANGLE header referred to the slit's physical position in the instrument, and after this date it referred to the slit's angle on the

sky. For data taken before this date, the header value internal to ORAC-DR is set to coorespond to the on-sky angle.
[CGS4/_FIX_SANGLE_HEADER_]

For UIST data taken before 2002 December 2 and Michelle, raw data units are converted from ADU per second to the UKIRT standard of total ADU per exposure.
[UIST/_DATA_UNITS_TO_ADU_, MICHELLE/_DATA_UNITS_TO_ADU_]

For Michelle there is a validation check of the waveform used, comparing the waveform name given in the headers with other metadata, and recipes issue a warning if there is an inconsistency. For lowQ and MedN2 data taken before 2004 March 09, the gratings were installed the wrong way round in the cryostat, so these data are flipped along the dispersion axis at this stage in the data reduction.
[MICHELLE/_CHECK_WAVEFORM_, MICHELLE/_FLIP_FLIPPED_GRATING_FRAMES_]

A night log is created or appended in `$ORAC_DATA_OUT` for each frame processed. This tabulates the main parameters of the observation having first corrected defective or undefined headers.
[_NIGHT_LOG_, <instrument>/_FIX_EXTRA_HEADERS_ for MICHELLE, CGS4, and UIST]

3.1.3 Bad pixels

The recipes apply a predetermined bad-pixel mask with the aim of removing the bulk of 'hot' and 'cold' pixels. This flags approximately 0.4% of UIST and ISAAC pixels, 0.1% of IRIS2 pixels, and 5% of Michelle's pixels.
[_MASK_BAD_PIXELS_]

Some of the instruments (UIST, CGS4) array tests are run, typically at the start of each night. As a part of these array tests a new bad-pixel-mask is generated on-the-fly, using the predetermined one as a basis. For UIST the new bad-pixel-mask is generated from a long-exposure dark observation, typically 100s. Any pixel that is 5-σ higher than the 3-σ clipped mean or 1000-σ lower than the 3-σ clipped mean is flagged as bad. For CGS4 any pixel higher than 1700 or lower than 15 for a dark whose exposure time is longer than 80 seconds, or higher than 1500 and lower than -100 for a dark whose exposure time is 80 seconds or shorter, is flagged as bad.

Creating a bad-pixel mask The easiest way to create your own bad-pixel mask for use with the calibration system, is to run the MAKE_BPM recipe on a long-exposure dark (at least 20 seconds integration). It is possible to change the symmetric σ-clipping bounds in the recipe (see primitive `_MAKE_BPM_BY_SIGMA_THRESHOLDING_`). You can tailor this primitive if you want more control, say to have asymmetric rejection or more sophisticated definitions.
[_MAKE_BPM_BY_SIGMA_THRESHOLDING_]

For better results, use the average of long dark frames taken across two or three nights. First, produce QUICK_LOOK versions of the long-exposure dark to flatten the NDF structure or convert the FITS file. Flag all pixels that are 5 standard deviations (σ) above and below the 3-σ clipped mean of the dark as "bad", then multiply the resulting frame by zero so that the resulting bad-pixel mask has data values of `0` and `bad` only. You can choose your own thresholds. Here is an example, using data from two nights of CGS4 data and Starlink software.

```
% oracdr_cgs4 20010101
% setenv ORAC_DATA_OUT `pwd`
% oracdr -list 4:4 QUICK_LOOK -nodisplay
```

```
% oracdr_cgs4 20010102
% setenv ORAC_DATA_OUT ‘pwd‘
% oracdr -list 4:4 QUICK_LOOK -nodisplay

% kappa
% add c20010101_00004_mraw c20010102_00004_mraw add_darks
% cmult add_darks 0.5 av_darks
% stats av_dark clip=3
% thresh av_darks av_darks_thresh -49 58 bad bad
% cmult av_darks_thresh 0 avbpm title=\"CGS4 bpm, January 2001\"
```

In the above example the 3-σ clipped mean was 4.27 and the standard deviation was 10.727, resulting in -49 and 58 as the lower and upper thresholds.

Then you specify the bad-pixel mask on the command line.

```
% oracdr -calib mask=avbpm ...
```

UIST has its own slightly different formula; see DARK_AND_BPM for details

[UIST/_FIND_BAD_PIXELS_, UIST/_FILE_BAD_PIXELS_, _FILE_MASK_]

3.1.4 Readnoise Variance

After the bad pixel mask has been applied, the readnoise variance is added into the VARIANCE component of the NDF. For all instruments, the readnoise value is obtained from the calibration system, having been previously calculted in the ARRAY_TESTS recipe. Since the readnoise value is stored in electrons, it must be converted into analogue-to-digital units.

[_ADD_READNOISE_VARIANCE_, _CALCULATE_NREADS_NOISE_FACTOR_]

CGS4 Readnoise For CGS4 the readnoise from the calibration system is first divided by a factor to take multiple non-destructive reads into account:

$$RNE = RN / \frac{3.8 + 3.5 * e^{-1/8 * N_{ND}}}{3.8 + 3.5 * e^{-1/8}}$$

where RNE is the readnoise per exposure, in electrons, RN is the readnoise in electrons, and N_{ND} is the number of non-destructive reads.

The formula used to determine the variance due to readnoise is:

$$V_{RN} = \frac{RNE^2}{N_e * gain^2}$$

where RNE is the readnoise per exposure, in electrons, N_e is the number of exposures per integration, and $gain$ is the detector gain in electrons per ADU.

Michelle Readnoise For Michelle the readnoise from the calibration system is also first divided by a factor to take multiple non-destructive reads into account:

$$RNE = \frac{RN}{\sqrt{\frac{N_{ND}*(N_{ND}+1)}{12*(N_{ND}+1)}}}$$

where RNE is the readnoise per exposure, in electrons, RN is the readnoise in electrons, and N_{ND} is the number of non-destructive reads.

The formula used to determine the variance due to readnoise is:

$$V_{RN} = \frac{RNE^2}{N_e * gain^2}$$

where RNE is the readnoise per exposure, in electrons, N_e is the number of exposures per integration, and $gain$ is the detector gain in electrons per ADU.

UIST Readnoise For UIST the formula used depends on the number of reads performed. For data taken with fewer than 13 reads, the formula used is:

$$V_{RN} = \frac{(RN * (-0.0322 * (N-1) + 1.0322))^2}{gain^2}$$

For data taken with between 13 and 51 reads the formula used is:

$$V_{RN} = \frac{(1.5616 * RN * ((N-1)^{-0.3568}))^2}{gain^2}$$

For data taken with more than 51 reads the variance due to readnoise is:

$$V_{RN} = \frac{225}{gain^2}$$

For all formulas RN is the readnoise in electrons, N is the number of reads, and $gain$ is the detector gain in electrons per ADU.

IRIS2 Readnoise For IRIS2 the formula used to determine the variance due to readnoise is:

$$V_{RN} = \frac{RNE^2}{N_e * gain^2}$$

where RNE is the readnoise per exposure, in electrons, N_e is the number of exposures per integration, and $gain$ is the detector gain in electrons per ADU.

ISAAC Readnoise ISAAC uses the same formulae for calculating the readnoise variance as Michelle *i.e.*

$$RNE = \frac{RN}{\sqrt{\frac{N_{ND}*(N_{ND}+1)}{12*(N_{ND}+1)}}}$$

and

$$V_{RN} = \frac{RNE^2}{N_e * gain^2}$$

where RNE is the readnoise per exposure, in electrons, RN is the readnoise in electrons, and N_{ND} is the number of non-destructive reads, N_e is the number of exposures per integration, and $gain$ is the detector gain in electrons per ADU.

3.1.5 Bias Subtraction

For observations not taken in non-destructive read mode, a bias frame is subtracted. The bias frame is pulled from the calibration system, having been filed using the REDUCE_BIAS recipe. *[_SUBTRACT_BIAS_]*

3.1.6 Poisson Variance

Once the readnoise variance has been added and the bias has been optionally subtracted, the variance due to Poisson noise is added. For all instruments the Poisson variance is calculated as:

$$V_P = S * gain * N_e$$

where S is the signal in ADU per exposure, $gain$ is the detector gain in electrons per ADU, and N_e is the number of exposures per integration.

At this stage the number of pixels that are background limited is displayed. This number is simply the percentage of pixels where the Poisson noise is greater than the readnoise. *[_ADD_POISSON_VARIANCE_]*

3.1.7 Chopping

In the thermal and mid-infrared regimes the sky is varying so rapidly that normal reduction methods are inappropriate. Instead sky subtraction is achieved either by frequently oscillating the secondary mirror between two beams (mid-infrared), called A and B; or by moving the telescope offsets (thermal) after a short exposure. The generic term is *chopping*.

Both methods produce frames with the target spectrum on different rows of the detector. The POINT_SOURCE and EXTENDED_SOURCE recipes difference these pairs of frames so that the result has both a positive and negative spectrum, and a background close to zero. The sense of the subtraction is always the same. ORAC-DR subtracts the B beam from the A beam, and the normal sequence is ABBA.
[_SUBTRACT_CHOP_]

3.1.8 Flat Fielding

Depending on the data format, this step and the subsequent step (interleaving and coadding, see Section 3.1.9

) may be swapped. If the flat frame and data frame were both taken with the same interleaving, then flat-fielding is done after interleaving. Otherwise, flat-fielding is done first. See Section **??**

for information on how spectroscopic flat-fields are created.

Flat-fielding is done by a straight division of the data frame by the appropriate flat-field calibration frame.
[_DIVIDE_BY_FLAT_, _FLATFIELD_COADD_INTERLEAVE_]

3.1.9 Interleave and Coadd

In order to fully sample a spectrum and reduce the effect of bad pixels, observations are often taken at different detector positions. The detector is stepped along the spectral axis by a fractional number of pixels; for CGS4this is typically in half-pixel or third of a pixel steps. Each spectral element can be sampled more than one time, which helps increase the signal-to-noise and decrease the impact of hot or bad pixels. This sampling method is often referred to as $2x2$ or $3x2$ sampling. The first number refers to the number of data points taken per resolution (or the inverse of the fractional pixel step size) and the second refers to the number of times each pixel has been observed. These observations need to be interleaved to create a higher-resolution spectral image.

The interleaving is done by expanding the input frames by the reciprocal of the fractional pixel step size, then blanking out the extra columns in these expanded frames. The origins are then shifted correspondingly, and the expanded frames are coadded together using the mean to create a final spectral image.
[_INTERLEAVE_COADD_, _FLATFIELD_COADD_INTERLEAVE_]

3.1.10 Orient Image Normally

Some instruments are set up such that the spectrum on the detector runs from higher wavelength to lower wavelength as pixel value in increased, which is reverse to expectations. This step flips the image so that the shorter wavelength is to the left and longer is to the right.
[_ORIENT_IMAGE_NORMALLY_]

3.1.11 Wavelength Calibrate

Wavelength calibration is necessary so that spectral features can be identified. ORAC-DRcurrently only calculates a wavelength estimation based on information contained in the FITS headers. It takes the values for dispersion and central wavelength, then applies these values as a linear wavelength scale to the spectral image.

This is often not good enough for accurate wavelength calibration as most grisms and dispersers produce higher-order dispersions, so manual wavelength calibration must be performed after spectra have been extracted.
[_WAVELENGTH_CALIBRATE_BY_ESTIMATION_]

3.2 Group Formation

After the individual frames have been processed, a composite group spectral image must be formed. Most infrared spectroscopic observations are taken in object-sky pairs, so the first step in group formation is subtracting the sky frame from its corresponding object frame.

3.2.1 Sky Subtraction

To be able to do sky subtraction, ORAC-DRneeds to know out of a pair of frames which is the object frame and which is the sky frame. To do this ORAC-DRexamines the FITS headers. If the offset for both right ascension and declination are less than 0.001 arcseconds, then the frame is treated as being on-source. Otherwise, the frame is off-source and is used as a sky frame.

IRIS2does not record telescope offsets for spectroscopy mode, so this method cannot be used. Instead ORAC-DRexamines the aperture used. If aperture A is used then the frame is on-source and the right ascension offset is set to zero, otherwise the frame is off-source and the right ascension offset is set to 26.92 arcseconds.

ESOinstruments are different again as observations are not done in object-sky pairs. Instead they are done in equal-sized blocks of object and sky observations, such that a certain number of object observations are done, followed by an equal number of sky observations. In this case the initial frame in a group is always assumed to be on-source. As with the standard pipeline, an observation is considered to be off-source if its offsets are greater than 0.001 arcseconds.
[_PAIR_REDUCTION_STEER_]

Sky subtraction is straightforward – the sky frame is subtracted from the object frame.

For ESOinstruments the corresponding sky frame in a block is subtracted from the respective object frame in a block, such that the same position in each set is considered as an object-sky pair.
[_PAIR_REDUCTION_SUBTRACT_]

3.2.2 Group Coaddition

Group coaddition is performed by taking the average of all of the sky-subtracted pairs in the group. The header values for airmass and UT time at the end of observation are updated in the group frame.
[_PAIR_REDUCTION_COADD_TO_GROUP_]

In polarimetry mode multiple group files are created, one for each waveplate position. Airmass and UT time header values are also updated.
[_PAIR_REDUCTION_COADD_TO_GROUP_POL_]

3.3 Spectrum Extraction

3.3.1 Counting Beams

The first step in doing spectral extraction is determining the number of beams to extract. For regular object-sky observations there will be one positive beam and, depending on how large the offsets are or if nodding was done along the slit or not, zero or one negative beams. A nod is considered to be along the slit if the nod angle is within 5 degrees of the slit angle, which

represents roughly 1 arcsecond over a 10 arcsecond throw. The length of the slit is not taken into account, so throws to a position off the end of the slit will still count as being along the slit, even though the spectrum will not appear on the detector.

For chopped observations there can be one or two positive beams and zero, one or two negative beams, depending on combinations of chop throw, chop angle, nod throw and nod angle. If the chop throw and nod throw are equal to within 2 arcseconds and the chop and nod are along the slit, there will be one positive beam and two negative beams. If the chop throw and nod throw are equal to within 2 arcseconds and the chop and nod are to sky, there will be two positive beams and two negative beams. If the chop is along the slit then there will be one positive beam and one negative beam. If the chop is along the slit but the offset is to sky, then there will be one positive beam and one negative beam. If the chop is to sky and the nod is along the slit, then there will be one positive beam and one negative beam. If the chop and nod are both to sky, then there will be one positive beam.

For dual-beam polarimetry observations the number of beams is as above, but doubled. For single-beam polarimetry the number of beams is as above.
[_EXTRACT_DETERMINE_NBEAMS_]

3.3.2 Finding Beams

After the number of beams to extract has been determined, it comes time to locate the beams on the detector. The spectral image first has any residual bias level removed by subtracting a multiply clipped mean, and it is then collapsed along the spectral axis to form a profile spectrum.

To find the beams, the profile spectrum is turned into a five-pixel wide image which is made up of the original profile flanked by symmetric half- and quarter-strength copies. This step is non-parametric, and can prefer faint blips over strong beams, although in practice the correct beam is found.
[_FIND_PEAKS_BY_MAKING_IMAGE]

If the number of beams found does not equal the number of beams calculated in the previous step (see Section 3.3.1

) then spectral extraction will not occur. If flux calibration is to be performed, then processing skips to division by standard (3.5), if division by standard and flux calibration is necessary.

IRIS2differs in that the entire spectral image is not collapsed to form the profile. Collapsing the entire image risks producing spurious peaks due to noisy data near the edges of the array, so a profile is formed by collapsing a region 0.05 microns short and 0.15 microns long of the central wavelength.

After the beam locations have been determined they are filed with the calibration system to be used for faint sources, if necessary.

The beam detection step described here does not modify the Group file.
[_EXTRACT_FIND_ROWS_]

3.3.3 Beam Extraction

Once the beam positions have been located, the beams can be extracted. First, an extraction window width is calculated based on the position and number of beams in the spectral image. This window is used for all beams. If there are two beams, then the half-width of the window

is half the beam separation. If there are three, then the half-width of the window is half the distance between the positive beam and the first negative beam. Otherwise, the half-width is 50 pixels.

UISTdiffers from this in that if either the HK or KL grisms are used, the half-width of the extraction window is 10 pixels.

For each beam, an optimal extraction profile (Horne, 1989) is then determined over the extraction window. If requested (i.e. if a standard star is reduced), this optimal extraction profile is filed with the calibration system.

The beams are then optimally extracted using the appropriate profile.
[_EXTRACT_ALL_BEAMS_]

3.3.4 Derippling

When observations are interleaved (see Section 3.1.9

), variations in throughput, sky brightness, and other factors may cause flux levels to change between the interleaved exposures, resulting in a ripple in the interleaved spectrum. This ripple is removed by creating a ripple flat with a ripple that has the same frequency as the interleaving, i.e. if 2x2 interleaving is performed, a ripple flat with a frequency of two pixels is created. This ripple flat is divided into the extracted spectrum unless the ripple amplitude is less than 70
[_DERIPPLE_ALL_BEAMS_]

3.3.5 Beam Cross Correlation

To remove any shift in beams, possibly caused by tilted spectra, the extracted beams are cross-correlated and shifted. If the maximum value of the cross correlation funtion is less than 0.6, or if the shift is greater than 2 pixels, then the spectra are not aligned and shifted.
[_CROSS_CORR_ALL_BEAMS_]

3.3.6 Extracted Beam Coaddition

The beams have been extracted and must now be coadded. This is done by simply averaging all of the extracted beams together.
[_COADD_EXTRACTED_BEAMS_]

3.4 Signal-To-Noise Calculation

When an extracted and co-added spectrum has been determined, the signal-to-noise ratio is calculated, but only if it has a variance array associated with it.
[_CALCULATE_SNR_]

3.5 Division By Standard Star

Whether or not a spectrum has been extracted, division by a standard star spectrum can still proceed. If no spectrum has been extracted, then the standard star spectrum is extended perpendicularly to the wavelength direction to make a 2-D spectrum.

When the spectrum of the science target is divided by a standard star spectrum, a straight division is done.

3.6 Flux Calibration

After standard division is done, the spectrum is flux-calibrated. This is done by using the standard star's V-band magnitude, spectral type, and observed waveband to determine a reference flux at the central wavelength of the observed waveband. The units of the flux-calibrated spectrum are presented in $W/m^2/\mu m$.

4 Science Target Recipe Details

There are three basic recipes available for reducing science target observations: EXTENDED_SOURCE, POINT_SOURCE, and FAINT_POINT_SOURCE. These three have many variants that dictate the use or lack of calibration data. For example, there exists a POINT_SOURCE_NOFLAT recipe that is meant to be used for point source reduction without flat-fielding. The use of these variants is typically avoided, as the basic recipes can often flag uncalibratable data.

In addition, dual-beam polarimetry can be reduced using the POINT_SOURCE_POL recipe. This recipe is a variant of the POINT_SOURCE recipe.

4.1 POINT_SOURCE

The POINT_SOURCE recipe is used to reduce observations of point sources. It reduces data using all of the steps outlined in Section 3

.

The following variants are available: POINT_SOURCE_NOARC (no arc spectrum extracted), POINT_SOURCE_NOFLAT (no division by a flat-field), POINT_SOURCE_NOFLAT_NOARC, POINT_SOURCE_NOFLAT_NOSTD, and POINT_SOURCE_NOSTD (no division by a standard star).

4.2 EXTENDED_SOURCE

The EXTENDED_SOURCE recipe is used to reduce observations of extended sources. It differs from the POINT_SOURCE recipe in that EXTENDED_SOURCE skips the spectrum extraction and proceeds directly with signal-to-noise calculation and division by a standard star .

The following variants are available: EXTENDED_SOURCE_NOARC (no arc spectrum extracted), EXTENDED_SOURCE_NOFLAT (no division by a flat-field), EXTENDED_SOURCE_NOFLAT_NOARC, EXTENDED_SOURCE_NOFLAT_NOSTD, and EXTENDED_SOURCE_NOSTD.

4.2.1 EXTENDED_SOURCE_WITH_SEPARATE_SKY

In addition to the variants to EXTENDED_SOURCE listed above, EXTENDED_SOURCE_WITH_SEPARATE_SKY exists for very extended sources where it is not possible to nod along the slit, thus a separate sky is used for sky-subtraction. The separate sky is reduced using the REDUCE_SKY calibration recipe.

The following variants are available: EXTENDED_SOURCE_WITH_SEPARATE_SKY_NOFLAT (no division by a flat-field), EXTENDED_SOURCE_WITH_SEPARATE_SKY_NOFLAT_NOSTD

(no division by a flat-field or by a standard star), and EXTENDED_SOURCE_WITH_SEPARATE_SKY_NOSTD (no division by a standard star).

4.3 FAINT_POINT_SOURCE

The FAINT_POINT_SOURCE recipe is used to reduce observations of point sources that are faint enough that they may not be detected in the beam detection step described in Section 3.3.2. In this case, the locations of the beams as determined for the standard star are used to extract the object. Optimal extraction is still used.

The following variants are available: FAINT_POINT_SOURCE_NOFLAT (no division by a flat-field), FAINT_POINT_SOURCE_NOFLAT_NOSTD (no fivision by a flat-field and no extraction of an arc spectrum), and FAINT_POINT_SOURCE_NOSTD (no extraction of an arc spectrum).

4.4 POINT_SOURCE_POL

Dual-beam spectropolarimetry data can be reduced using the POINT_SOURCE_POL recipe. This recipe reduces data using the steps in Sections 3.1, 3.2, and 3.3. The data can be taken in any order between waveplate positions, but corresponding object-sky pairs for a given waveplate position must be observed together. Spectrum extraction is done such that separate *E* and *O* beam spectra are created for each waveplate position. After a cycle of eight observations has done, corresponding to one object-sky pair at each of the 0-, 45-, 22.5-, and 67.5-degree waveplate angles, the pipeline calculates the *I*, *Q*, and *U* Stokes parameters, from which the percentage polarization, polarization intensity, and polarization angle spectra are calculated.

5 Calibration Recipe Details