**long_coadd2d**

Developed by R. da Silva and M. Fumagalli

Last updated October 2, 2009

# Contents



# 1 Introduction: When to use it

The Low-Redux pipeline develeoped by Joe Hennawi, Scott Burles, David Schlegel, and J. Xavier Prochaska is a robust data reduction pipeline that can reduce data from some of the world's top observatories. However it is optimized for reducing single exposures of point source spectra. If you are interested in coadding multiple exposures or measuring surface brightness in a given spectral range you must use the Low-Redux extension long_coadd2d. For more information look at the documentation for long_coadd2d which can be found by typing the following at the IDL command line:

doc_library, 'long_coadd2d'

---

# 2 Coadding Spectra

## 2.1 How to use it

The first step is to run your data throught the Low-Redux pipeline. For instruction on how to do this, please visit http://www.ucolick.org/~xavier/LowRedux/index.html .

Long_coadd2d takes at a minimum 2 pieces of information: the names of the science files output from Low-redux and trace index. The first should be trivial, it is simply a string array of the locations of the output science files that you wish to coadd and/or flux. The first filename in the array will be used as the reference. The trace index is slightly more complicated. The 5th extension[1] of Low-Redux's science image outputs (sci-***.fits.gz) are arrays of structures relating to the extraction. You will need to examine these structures to see which index of this structure corresponds to the trace that you will want to use as your reference. Suggestions are to look at the XPOS tag as that denotes the x-position of each trace. Also you can make plots of WAVE_OPT vs. FLUX_OPT to see the extracted flux from each object. The current version of the program will only work if you have an object bright enough in the slit to be traced by Low-Redux. See below for an example:

long_coadd2d, ['sci-b116.fits.gz', 'sci-b117.fits.gz'], [0, 3]

Here the list of science files is ['sci-b116.fits.gz', 'sci-b117.fits.gz'] and the trace indices are [0, 4]. This means that the 1st file in the structure array which is the 5th extension of sci-b116.fits.gz corresponds to the same object as the 4th object in the 5th extension of sci-b117.fits.gz. We start counting at 0. Note that the trace index is *not* always the OBJID This command will produce a file called coadd***.fits.gz. This file contains several extensions

**0:** The stacked sky-subtracted image

**1:** The inverse variance of the stacked image. Flagged pixels are set to 0.

**2:** A structure containing information about the stacking

**3:** A stacked mask. Here pixels are only zero if there is no flagged pixel at that position in any of the images.

**4:** The wavelength image. This image has been properly adjusted for flexure and heliocentric corrections (see below) and corresponds to the coadded spectrum. This extension is only written if you supply the wvimg keyword (see below). The wavelengths are given in vacuum Angstroms

This file will also output a log file of the same name as the main output file but with the .log suffix. This log file details the actual programming running.

## 2.2 Optional Keywords

### Common

**/integer** set if you only want to shift by an integer pixel amount. This reduces correlated noise but will introduce biases in velocity and spatial extent

[1]You can access the the 5th extension of a fits file with mrdfits as follows: extension5=mrdfits('file.fits', 5)

/**cr_reject** set to use information of multiple exposures to deal with cosmic rays. *Recommended if you have 3 or more exposures.* This keyword rejects pixels to the coaddition that are $10\sigma$ away from the median of the images. Here $\sigma$ is the error in that pixel and NOT the standard deviation of the images.

**weights** set to an arbitray set of scalar weights for each image (must be an array with a number of elements equal to the number of input images)

**Wavelength**

**wvimg** set to the wavelength image wave-***.fits output by Low-Redux. If set this will add an output extension to the output file. This keyword is mandatory for any type of fluxing.

/**nohelio** set to suppress heliocentric correction to wavelength image. At this point in time the code only applies a single heliocentric correction which is the average of all the heliocentric corrections. Thus it should not be trusted for images spanning large temporal ranges. The log file outputs the heliocentric correction for each image. Use your judgment relevant to your science to determine if this is a problem for your data.

**Less-Common**

/**xwave** set if your x-axis is your science files x-axis is the wavelength axis (most likely does not ever need to be set... mainly a developer keyword)

**outputname** if set, its value replaces the default name of the output file [must be a string]

**splogname** if set, its value replaces the default name of the log file [must be a string]

/**quiet** if set, then supresses the log output to the command line

**stacked_mask** the same stacked mask as the 3rd extension of the output file may be obtained through this keyword as well

/**no_fwhm_weight** set if you do not wish to weight by FWHM of your trace

## 2.3 What it does

First it extract necessary information from science files. Using the first image given in your list of science images as a reference, the code determines the spatial shift by looking at the difference in the medians of the XPOS tag's x-position. The dispersion axis difference is found by looking at the difference in the FLX_SHFT_WAV tag of the science file which notes the dispersion axis's flexure. These quantities were all previously computed by Low-Redux. If the /**integer** keyword is not set, the code performs a sigma filtering to remove cosmic rays through **sigma_filter.pro**. All replaced pixels are flagged. Then all flagged pixels (flagged by Low-Redux or **sigma_filter.pro**) are then interpolated over. The shifts are applied with **shiftf.pro**. Flagged pixels and pixels that have interpolated values from flagged pixels are flagged in the new shifted image.

The images are then coadded according to

$$Image_{x,y} = \frac{\sum_i w_i scale_i image_{x,y,i}}{\sum_j w_j}$$

$$\sigma_{x,y} = \sqrt{\frac{\sum_i w_i^2 scale_i^2 \sigma_{x,y,i}^2}{\sum_j w_j^2}}$$

Where default values of the scales are taken to be the ratio of the medians of the FLUX_OPT extractions to the reference exposure. The weights are taken to be the median inverse variance for each sky-subtracted image. By default the weights are taken to be the median FWHM of the extractions. Additional weights can be supplied via the **weight** keyword. Additional weigths are muliplied by the default weight.

Note that our error bars are taken to assume no covariance of which we know that some exists (since we interpolate). Thus these error bars are only estimates, but should be reasonable.

---

# 3 Two-dimensional Fluxing

## 3.1 How to use it

In order to perform a 2-dimensional fluxing such that you will receive an output image in units of $10^{-17}$ $ergs/s/cm^2/Ang/arcsecond^2$, you simply need to set the following keywords

**/flux2d** must be set so that the calculation will be performed

**wvimg** must be provided and set a string that details the location of the output wave image from Low-redux

**pixscale** the pixel scale of your image must be set in arcsecond/pixel. Don't forget to account for binning! (i.e. a spec sheet may say .135 arcseconds/pixel but if you bin by 2 in the spatial direciton you need to put in .27)

**slitwidth** the width of your slit in arcseconds (default to 1)

**sensfuncfile** the sensitivity function file output by **long_sensfunc()**. See details at Low-redux webpage

As an example see below:

long_coadd2d, [‘sci-b116.fits.gz’, ‘sci-b117.fits.gz’], [0,3],/flux2d,wvimg=‘wave-b100.fits’,

pixscale=.211, slitwidth=1, sensfuncfile=’sens.fits’

Files are output to flux2d***.fits with the following extensions

**0:** fluxed image

**1:** fluxed inverse variance image

**2:** the two-dimensional sensitivity funciton defined as follows

$$Fluxed\ image = unfluxed\ image \times 2d\ sensitivity\ function$$

### 3.2 What it does

This program uses the one-dimensional sensitivity function from **long_sensfunc()** and applies it to the entire two-dimensional image by using the wavelength image and the sensitivity functions mapping between $\lambda$ and response. It does not take into account spatial variation of the sensitivity function but does take into account the different wavelength mapping for each column. We note that this is not an ideal solution. The correct operation would involve a requirement of multiple standard stars taken at different positions on the CCD. However in general the CCD response does not vary widely across its spatial direciton. A good sanity check is to look at how flat your flat fields are in the spatial direction.

---

## 4 One-dimensional Extractions and fluxing

### 4.1 How to use it

In order to extract you must set the /**extract** keyword and to flux you must include the **wvimg** and **sensfuncfile** keywords after you set /**onedflux**. We note that you can /**extract** without /**onedflux** but you cannot /**onedflux** without /**extract** since the /**onedflux** acts on the output from the /**extract** command.

long_coadd2d, [‘sci-b116.fits.gz’, ‘sci-b117.fits.gz’], [0,3],/extract,wvimg=‘wave-b100.fits’,

/onedflux, sensfuncfile=’sens.fits’

Extraction files are saved to extr***.fits and fluxed extractions are saved to flux1d**.fits The extr***.fits file has the following tags (ignore other tags):

**OBJID** Identifier noting the object

**SLITID** identifier for slit

**MED_SN2** median signal to noise squared

**FWHM** median FWHM of the optimal extraction fit

**FWHMFIT** FWHM of each spatial pixel

**PROFILE** 2-dimensional fit to the spectrum. Normalized to 1 in the spatial direction denoting the spatial distribution of the spectrum

**WAVE_OPT** 1-dimensional wavelength that corresponds to optimal extraction [Ang]

**FLUX_OPT** 1-dimensional flux that corresponds to optimal extraction [$10^{-17} ergs/s/cm^2/Ang$]

**IVAR_OPT** 1-dimenstional inverse variance of optimal extraction

**WAVE_BOX** 1-dimensional wavelength that corresponds to boxcar extraction [Ang]

**FLUX_BOX** 1-dimensional flux that corresponds to boxcar extraction [$10^{-17} ergs/s/cm^2/Ang$]

**IVAR_BOX** 1-dimensional inverse variance of optimal extraction

To make a 2 dimensional image profile (for subtraction purposes) you need to do the following():

```
extr=mrdfits('extr***.fits', 1)

dims=size(extr.profile, /dim)

model =extr.profile*(extr.wave_opt##replicate(1,dims[0])
```

### 4.2 What it does

This code uses Low-Redux's extraction programs on the coadded data and then fluxes it as **long_fluxcal** does. Units of flux are $10^{-17}$ $ergs/s/cm^2/Ang$.