GHOST Programmer's Manual

Release 0.9

TBD

May 19, 2017

Contents

CHAPTER 1

Introduction to the Instrument

General Description

Description of the Modes

Required Calibration and Associated Observations

Important Instrument Characteristics and Issues

CHAPTER 2

Typical Data Reduction Flows

List of Typical Sequences

Top-Level Flow Charts for Processing of Calibrations

Top-Level Flow Charts for Processing of Science

CHAPTER 3

AstroData Types

Relevant AstroData Types

list all astrodata types to be considered

Association Table

associate ad types with type of observations

CHAPTER 4

Recipes and Contexts

Contexts

list the contexts. Typically those are QA, QL, and SQ, for Quality Assessment, QuickLook, and Science Quality, respectively.

Recipes

list of recipes for each context

location of recipes and indexes

Technical Flow Charts

technical flow charts for each recipes

Issues and Limitations

CHAPTER 5

Primitives

CHAPTER 6

Test Suite

Available Tests

Missing or Desirable Tests

Running the Tests

CHAPTER 7

Configuration

This chapter details configuration procedures that will almost always be undertaken by Gemini Observatory staff.

Generating New `Polyfit` Models

Note: This is the second iteration of this documentation.

This is necessary if the instrument is realigned at any point during its lifetime at the telecope. Much of this process is common with the normal data reduction, with the exception of the manual model adjustment.

The principle behind this process is that the format of the spectral orders and the wavelength scale can be modelled uniquely using polynomials of polynomials. e.g. A series of polynomials as a function of order number are combined in a polynomial fashion as a function of y position on the chip.

In an ideal world, there would only ever be one `polyfit` model for each configuration (i.e. arm and resolution) of the instrument, which can be determined during instrument commissioning and used forever after. However, it cannot be guaranteed that the instrument will remain stable for its entire lifetime. For example, minor earthquakes may cause optical components to shift ever so slightly, or maintenance may require disassembly of the instrument, which may alter the position of apertures on the detector.

The process to generate a new `polyfit` configuration is done entirely within Python. A subset of these `polyfit` methods are used in the `findApertures` primitive in `astrodata_GHOST` to fit the models to observational data.

The `polyfit` module is used at this stage and requires knowledge of the spectrograph arm, mode and a reduced flat field image (tested only with flats from the simulator).

Usage follows:

```
import polyfit
import astropy.io.fits as pyfits
ghost = polyfit.ghost.Arm('red',mode='high')
```

At this stage it is important to an initial guess array for the polynomial model.

At this point a location must be defined and the data imported (e.g.):

```
model_file='~.local/lib/python2.7/site-packages/ghostdr-0.1.0-py2.7.egg/astrodata_
↪GHOST/ADCONFIG_GHOST/lookups/GHOST/Polyfit/red/161120/high/xmod.fits'
xparams=pyfits.getdata(model_file)
```

After acquiring the flat field data:

```
flat_file = "location_to_flat/flatfield_frame.fits"
flat_data = pyfits.getdata(flat_file)
```

a convolution map is required. This is done so that, irrespective of the number of fibers per order, the model is adjusted against an equivalent map that has maxima where the middle of the order lies.

Either a supplied model of the slit profile (from the slit viewer) or a default uniform illumination profile is convolved with every column of the flat field image along the spatial direction, resulting in a series of images that match the centers of the orders.

This is then fed into the `adjust_model` function for visual inspection of the initial model:

```
flat_conv=ghost_format.slit_flat_convolve(flat_data)
adjusted_xparams=ghost_format.adjust_model(flat_conv,xparams=xparams,convolve=False,
↪percentage_variation=10)
```

The `percentage_variation` refers to the percentage range of values that each parameter is allowed to be varied by the matplotlib slider widgets. This is set at 10 percent by default and should be enough for small changes. Increase if the large adjustment is needed.

If this is performed because of a recent alignment of the instrument, then the `Submit` button saves the current version of the model onto the calibrations directory. The `adjust_model` function is for engineering use only at this stage and the user should not have to see it.

Once the model is close, the model can be fitted:

```
fitted_model=ghost_format.fit_x_to_image(flat_conv,xparams=adjusted_xparams, decrease_
↪dim=8,inspect=True)
```

This function takes the convolution map and adjusts the model to the local maximum along each order. The `inspect` parameter set to True displays the result of the fit.

At this point, the `fitted_model` variable contains the result of the fit and should be used as the input to the flux extraction code. This new 2D array should overwrite the current default xmod file for this arm/mode in the correct location. At this point it is important to also place it in whatever location has been agreed so that the pipeline uses whichever model is appropriate for any date.

Python Module Index

a

Index

A