Goodman HTS Pipeline User Manual

version 0.1

Simon Torres

June 30, 2017

Contents

Goodman High Throughput Spectrograph Pipeline - User Manual

This document is the user manual for Goodman HTS Pipeline

Overview

What is it?

This is a set of tools for data reduction of the Goodman High Throughput Spectrograph's data. This instrument is currently in operation at Soar Telescope in northern Chile.

Although we have been trying to make the pipeline as *flexible* as possible and although it has a high level of automatization is important that you follow our *Observing Guidelines* while obtaining the data. Those guidelines consider what we believe it will work best for most scientific programs. For instance, if you are doing radial velocity studies you will need to have, ideally, the science target *"bracketed"* by comparison lamps or if you don't care much about radial velocity precision you would apply one single wavelength solution to all the data obtained throughout the night.

The software is developed in Python 2.7. We use conventions and tools to improve efficiency of development, such as. *Python Enhancement Proposal* (PEP8, PEP256), GitHub, sphinx, etc. The goal is to have a well-written well-documented code so that anyone could jump in to contribute as developer. We also kept in mind scalability.

Exactly, how it works?

It is composed of two main pipelines to process *images* and a second one for spectroscopic processing. At the moment they are not integrated into a single tool and you have to call them separately. Therefore, the full processing of an spectrum is split in several subprocesses but we can group them in four main groups: *Image Reduction*, *Identification and Extraction*, *Wavelength Calibration* and *Flux Calibration*. At the moment we haven't considered to implement a photometric nor an astrometric module.

Image reduction: redccd

This is a common process for all the data, be it for *imaging* or for *spectroscopy*. The steps are the following:

1. Clean Path: You **MUST use different directory for input and output** with *redccd* or you may end up with your data deleted. But don't worry, the program doesn't allow it. Although you are free to use any other directory, the recommended way is as follows:

   ```
   $ redccd [options] /source/path/ /source/path/RED/
   ```

2. Fix all headers and change image dimensions. Goodman's Blue Camera contains non-ASCII characters on its headers, therefore they need fixing otherwise the next processes will crash.
3. Create a table with relevant information of the files present on the source directory and classify them accordingly.
4. Creates Master Flats, Master Bias, process Night Flats.
5. Reduce Arc and Science Frames.

Finally it delivers reduced images with inside `/source/path/RED` with a prefix currently set to `fzh_` where **f** stands for *flat corrected*, **z** for *zero* or *bias corrected* and **h** for header corrected and data resized.

Spectroscopic Reduction: redspec

The spectroscopic reduction process is more complicated and can be split in many subprocesses. Here you will only find a summary, if you want detailed information please visit the code documentation page. The steps are the following:

1. Identify spectroscopic targets in an image. This is done by looking for peaks in the spatial direction:

2. Trace. Once it has a list of potential targets it will try to find the trace of the spectrum, in case it fails it will discard the target, if not, will provide the trace of the spectrum. The target is discarded also if the trace is too misaligned.
3. Extract. It will check the number of targets first and then define extraction zones for the spectrum itself and for the background subtraction regions. Spectrum extraction zones have have the highest priority. By default there are two background extraction zones but in case the positioning ends up near a neighbouring spectrum this background extraction zone will be discarded and only one will be used.
4. Wavelenght Calibration. At the moment is only possible to do interactive wavelength calibration but we are working on an automatic wavelength calibration module.
5. We have plans to implement in the (very) near future a flux calibration module.

Observing Guidelines

In order to obtain optimal results and be able to use our pipeline we recommend the following procedure:

Installing

Importing Parts of the Code

You can import almost every function in this pipeline:

```
1 from goodman_ccd.core import spectroscopic_extraction
2 # do some stuff here
3 print('Hello World')
```