Goodman HTS Pipeline User Manual

version 0.1

Simón Torres

August 02, 2017

Contents

Introduction

This document is the User Manual for the *Goodman Spectroscopic Data Reduction Pipeline*. It provides an overview of the main features of the pipeline, instructions on its use and how to run it on our dedicated *SOAR Data Reduction Server*, and installation instructions for those who wish to run it on their own computers.

Overview

The Goodman Spectroscopic Data Reduction Pipeline - GOODSPEC - is a Python-based package for producing science-ready, wavelength-calibrated, 1-D spectra. The goal of **goodspec** is to provide SOAR users with an easy to use, very well documented software for reducing spectra obtained with the Goodman spectrograph. Though the current implementation assumes offline data reduction, our aim is to provide the capability to run it in real time, so 1-D wavelength calibrated spectra can be produced shortly after the shutter closes.

The pipeline is primarily intended to be run on a data reduction dedicated computer. Instructions for running the software are provided in the Using Pipeline section of this guide. The Goodman Spectroscopic Data Reduction Pipeline project is hosted at GitHub at this link. Currently the pipeline is separated into two main components. The initial processing is done by `redccd`, which trims the images, and carries out bias and flat corrections. The spectroscopic processing is done by `redspec` and carries out the following steps:

- Identifies multiple targets (spectra of more than one object in the slit)
- Trace the spectra
- Extract the spectra
- Do background subtraction
- Find the wavelength solution. Defaults to automatic wavelength solution, but can be done interactively
- Linearize data (resample)
- Write wavelength solution to FITS header
- Create a new file for the wavelength calibrated 1D spectrum

Features

- Self-contained, full data reduction package for the most commonly used spectroscopic setups with Goodman. Given the almost limitless number of possible configurations avaliable with the Goodman instrument, only the most popular configurations will be supported, though we will try to add as many modes as possible.
- Python based, using existing Astropy libraries as much as feasible.
- Extensively documented, using general coding standards: PEP8 – Style Guide, PEP257 – Docstrings Convention (in-code documentation) – Google Style
- Multiplataform compatibility (tested on Linux Ubuntu, CentOS and MacOSX).
- Modular design. Could be used as a library within other Python applications.

Ways to run the pipeline

There are two ways to use the pipeline.

1. **Run it directly on a SOAR data reduction server** that you can access using VNC.
2. **Download and install the pipeline** (go to the Install section of this manual). Though we will try our best to provide answers to quick and simple installation issues, we cannot provide general installation support.

What the pipeline does not do

- In its current version the pipeline does not perform combination of individual spectra. If you obtained several individual exposures of the same object, they will be output as separate 1-D, wavelength-calibrated spectra
- There is yet no flux calibration. We are working on a module that will do this.
- The pipeline does not do miracles. If your have saturated flats, or your flats were taken with a different slit than your science frames, or you have no bias frames, the pipeline will fail.

General Considerations on using the pipeline

The Goodman Spectroscopic Pipeline is meant to work as a single package. However, the full process is split in two separate modules: `redccd` and `redspec`. The first does the basic 2D image reduction, applying bias and flat field corrections, and cosmic ray removal. The second module, `redspec`, takes the corrected 2D images output by `redccd` and produces wavelength-calibrated 1D spectra.

The pipeline is run from the command line in a terminal window. Each module is run separately, first `redccd` followed by `redspec`, however, you could run both sequentially from e.g. a shell script.

In order to facilitate things you should organize your data:

1. Make sure all the data in your folder corresponds to the same binning, readout mode, region of interest (ROI), and grating/wavelength mode combination.
2. You should have bias, flats (quartz or dome flats), and the appropriate comparison lamps. Other files like acquisition images, slit images and focus images should be deleted.
3. Do not mix dome flats with quartz lamp flats. As an example, suppose I took both quartz lamps and dome flats for my targets. I could create two folders, one with the science data and the dome flats, and another with the same science data and the quartz lamps. Then, if I run the pipeline in each folder I can compare the results and decide which type of flat works best for my particular case.

Command line arguments

For a list of the options and command line arguments type `--help` argument:

For `redccd`

```
usage: redccd [-h] [--auto-clean] [--cosmic <method>]
              [--dcr-par-dir <dcr.par_directory>] [--debug]
              [--flat-normalize <normalization_method>]
              [--flat-norm-order <order>] [--ignore-bias] [--ignore-flats]
              [--keep-cosmic-files] [--log-file <log_file>]
              [--raw-path <raw_path>] [--red-path <red_path>]
              [--saturation <value>]

Goodman CCD Reduction - CCD reductions for Goodman spectroscopic data.

optional arguments:
  -h, --help            show this help message and exit
  --auto-clean          Automatically clean reduced data directory
  --cosmic <method>     Clean cosmic rays from all data. Options are: 'dcr',
                        'lacosmic' or 'none'. Default is 'dcr'. See manual for
                        full description of dcr.
  --dcr-par-dir <dcr.par_directory>
                        Directory of default dcr.par file
  --debug               Show detailed information of the process.
  --flat-normalize <normalization_method>
                        Choose a method to normalize the master flat
                        forspectroscoy. Choices are: mean, simple (model) and
                        full (fits model to each line).
  --flat-norm-order <order>
                        Defines the order of the model to be fitted. Default
                        to 15
  --ignore-bias         Ignore bias correction
```

```
  --ignore-flats        Ignore flat field correction
  --keep-cosmic-files   After cleaning cosmic rays with dcr, do not remove the
                        input file and the cosmic rays file.
  --log-file <log_file>
                        Name for log file. Default name is goodman_ccd.log.
                        The file is written in <red_path> and will be deleted
                        each time you run this program
  --raw-path <raw_path>
                        Path to raw data.
  --red-path <red_path>
                        Path to reduced data.
  --saturation <value>  Saturation limit. Default to 65.000 ADU (counts)
```

And for `redspec`

```
usage: redspec [-h] [--data-path <Source Path>]
               [--proc-path <Destination Path>]
               [--search-pattern <Search Pattern>]
               [--output-prefix <Out Prefix>] [--extraction <Extraction Type>]
               [--reference-files <Reference Dir>] [--interactive] [--debug]
               [--log-to-file] [--max-targets <max targets>] [--save-plots]
               [--plot-results]

Extracts goodman spectra and does wavelength calibration.

optional arguments:
  -h, --help            show this help message and exit
  --data-path <Source Path>
                        Path for location of raw data. Default <./>
  --proc-path <Destination Path>
                        Path for destination of processed data. Default <./>
  --search-pattern <Search Pattern>
                        Pattern for matching the goodman's reduced data.
  --output-prefix <Out Prefix>
                        Prefix to add to calibrated spectrum.
  --extraction <Extraction Type>
                        Choose a which extraction to perform. Simple is a sum
                        across the spatial direction after the background has
                        been removed. Optimal is a more advanced method that
                        considers weights and profilefitting.
  --reference-files <Reference Dir>
                        Directory of Reference files location
  --interactive         Interactive wavelength solution.Disbled by default.
  --debug               Debugging Mode
  --log-to-file         Write log to a file
  --max-targets <max targets>
                        Maximum number of targets to be found in a single
                        image. Default 3
  --save-plots          Save all plots in a directory
  --plot-results        Show wavelength calibrated spectrum at the end.
```

Running the pipeline in the SOAR data reduction computer

The Goodman Spectroscopic Data Reduction Pipeline has been installed on a dedicated computer at SOAR. The procedure is to open a VNC session, for which you need to be connected to the SOAR VPN. The credentials for the VPN are the same you used for your observing run, provided by your *Support Scientist*, who will also give you the information for the data reduction computer VNC connection.

Establish a VNC connection

For the rest of this tutorial we will assume your host name is `vnc-server` the display is `1` and your password is `password`. Though we recommend using RealVNC, most other VNC clients will work fine (e.g., Remmina in Linux). For GNU/Linux and Mac OSX machines we suggest the RealVNC Viewer client. For Windows machines, we suggest either the RealVNC Viewer client or the UltraVNC viewer client. We also know that Vinagre and vncviewer on GNU/Linux work fine.

VNC from the Terminal

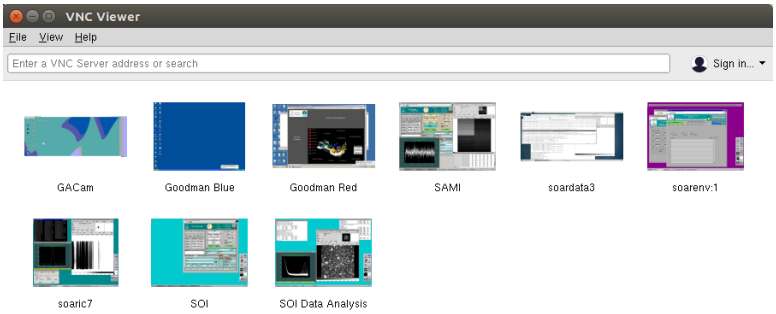
Open a terminal, and assuming you have installed `vncviewer`.

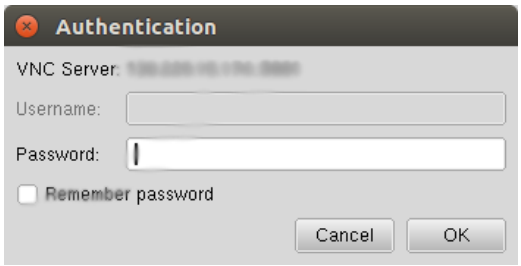
```
vncviewer vnc-server:1
```

You will be asked to type in the *password* provided above.

VNC using a Graphical Client

Using a graphical VNC client is quite similar and intuitive





In this case the *IP address* was used, which is equivalent and sometimes better.

Running the Pipeline

1. Open a Terminal
2. Go to `/home/goodman/data`

   ```
   cd /home/goodman/data
   ```
3. Here you have a workspace to put your data according to your institution.

4. Create a data folder inside your workspace.

```
cd NOAO
mkdir 2017-07-05
cd 2017-07-05
```

5. Copy your data from Goodman Computer

```
scp observer@soaric7:/home3/observer/GOODMAN_DATA/NOAO/2017-07-05/ ./
```

6. Make sure you have a full data set. At this point your observing logs will become very useful, eliminate focus sequence, aquisition exposure and any other file present that will not be needed for the processing. The following list summarizes the kind of data that you need to fully process your data.

    - BIAS: Bias
    - FLAT: Flats
    - COMP: Comparison Lamps
    - OBJECT: Science Frames

   Also make sure your data has the same *readout speed*, *binning*, and *ROI*. If you used different configurations during the same night, we recommend you to set up a separate folder for each.

7. Run `redccd`:

   For `redccd` I suggest using `--cosmic` and `auto-clean` also you might want to consider `--saturation <new value>` to change the saturation level if you get all your flats rejected due to saturation. Sometimes there is a hot column at the end that produced very high values.

   ```
   redccd --cosmic --auto-clean
   ```

   In case you want to use `--saturation` here is an example:

   ```
   redccd --cosmic --auto-clean --saturation 70000
   ```

   This changes the saturation level to *70000 ADU*` in this context the saturation value works as a threshold for rejecting images.

   By default, `redccd` puts reduced data in a subdirectory `RED`, you can provide a different one by using `--red-path`.

   An image `image_file.fits` that has been fully (and propperly) processed should have the new name (including the reduced data folder):

   ```
   cfzsto_image_file.fits
   ```

   Where `c` stands for *cosmic ray rejected*, `f` for flatfielded, `z` for zero or bias corrected, `s` for slit trimmed, `t` for trimmed and `o` for overscan corrected.

8. Run `redspec`:

   By default `redspec` will search for images with the prefix `cfzsto`, in case you have produced a different prefix you can change it by using `--search-pattern`

   You can just run `redspec` in case everything is the default but if this is the first time you run the pipeline I suggest:

   ```
   redspec --plot-results
   ```

   In that way two important plots will be shown full screen, the comparison lamp fitted to a reference comparison lamp and some values for the wavelength solution fit and the extracted spectrum plotted with the wavelength solution.

Troubleshooting

- The wavelength Solutions is way off: Check that the lamp was correctly registered in the header. Also check that the corresponding reference lamp exist. for instance is not the same to have `HgArNe` to `HgAr`
- Can’t detect any objects: Check that the keyword `OBSTYPE` is correct.

Installation Instructions

Installation will slightly depend on the system but in general is simple and can be sumarized in the following steps:

- Download the pipeline code
- Install prerequisites
- Install the pipeline

Install Dependencies

There are two types of dependencies that have to be met. The system prerequisite installation depends on the platform itself so they will be detailed bellow in their respective subsection.

The python libraries are specified in the file `requirements.txt` and installing them is very easy. But it has to be done later on.

Ubuntu 16.04

First install `git` that will let you clone the repository from GitHub.

```
sudo apt-get install git
```

Some other python-specific tools, if you already run python code most likely you already have them.

```
sudo apt-get install python-setuptools python-dev build-essential
sudo easy_install pip
```

We have decided to use Qt4Agg backend since Qt seems to be the most multi platform compatible backend.

```
sudo apt-get install python-qt4
```

CentOS 7

Install `git` if you don't have it already.

```
sudo yum install git
```

Then you need to install the `EPEL repository`

```
sudo yum -y install epel-release
```

Update the database with

```
sudo yum -y update
```

This takes a while

Install pip

```
sudo yum -y install python-pip
```

Upgrade pip

```
sudo pip install --upgrade pip
```

Install python-devel

```
sudo yum install python-devel
```

Installing on MacOSX

MacOS X intallation has not been fully tested.

Install Using Virtual Environments

Downloading the Goodman HTS Spectroscopic Pipeline

In order to get the code for the pipeline there a many options, our suggestion is to download the official release `tar.gz` file

Also you can clone it from GitHub

```
git clone https://github.com/soar-telescope/goodman.git
```

Although we will not provide support.

Installing the Pipeline

First of all install the python requirements. Your location must be the same as the file `requirements.txt` which should be your recently cloned repository

```
sudo pip install -r requirements.txt
```

Once this has succeeded proceed to install the pipeline using:

```
sudo python setup.py install --record files.txt
```

This will install the pipeline in your system and also will create a file `files.txt` that contains the list of files created at installation time and will be very helpfull if you ever want to fully remove the pipeline.

Install DCR

In terms of cosmic ray rejection we shifted to a non-python package because the results were way better compared to LACosmic's implementation in astropy. LACosmic was not designed to work with spectroscopy though.

Visit this Link to download the code and find the instructions for compiling. I have added a few pre-compiled binaries and if you are lucky they will work right way. The available binaries are located in `goodman/dcr` and the options are:

- dcr.Ubuntu16.04
- dcr.Centos7
- dcr.MacOSSierra
- dcr.Solaris11

Choose whatever version fits your needs and rename it `dcr` and put it in a folder that at the same time is in your `$PATH` variable. If you don't know what that is follow the next section.

Install binary DCR

1. Open a terminal
2. In your home directory create a hidden directory `.bin` (Home directory should be the default when you open a new terminal window)

   ```
   mkdir .bin
   ```
3. Move the binary of your choice and rename it `dcr`. If you compiled it most likey it's already called `dcr` so you can ignore this step.

   ```
   mv dcr.Ubuntu16.04 ~/.bin/dcr
   ```
4. Add your `$HOME/.bin` directory to your `$PATH` variable. Open the file `.bashrc` and add the following line.

   ```
   export PATH=$PATH:/home/myusername/.bin
   ```

   Where `/home/myusername` is of course your home directory.
5. Close and reopen the terminal or load the `.bashrc` file.

   ```
   source ~/.bashrc
   ```