

Goodman HTS Pipeline User Manual

version 0.1

Simón Torres, César Briceño and Bruno Quint

January 08, 2018

Contents

| | |
|---|-----------|
| Introduction | 1 |
| Overview | 1 |
| Features | 1 |
| Ways to run the pipeline | 1 |
| What the pipeline does not do | 2 |
| General Considerations on using the pipeline | 3 |
| Command line arguments | 3 |
| Lists of Reference Lamps Available | 5 |
| Adding new reference lamps | 5 |
| Running the pipeline in the SOAR data reduction computer | 6 |
| Establish a VNC connection | 6 |
| VNC from the Terminal | 6 |
| VNC using a Graphical Client | 6 |
| Running the Pipeline | 7 |
| Using the Interactive Mode | 9 |
| Troubleshooting | 16 |
| Installation Instructions | 17 |
| Anaconda | 17 |
| Install DCR | 18 |
| Install binary DCR | 18 |

Introduction

This is the User Manual for the *Goodman Spectroscopic Data Reduction Pipeline*. It provides an overview of the pipeline's main features, 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 [it's GitHub Repository](#).

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
- Estimate and subtract background
- 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 available 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
- Multiplatform 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.
- This pipeline does not evaluate nor select data by quality. It will simply try to run using all existing files. **Make sure you only have good data in the folder that will be reduced.**

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 make things easier 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 you took both quartz lamps and dome flats for your targets. You 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 you run the pipeline in each folder you 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
                  forspectroscopy. 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
```

General Considerations on using the pipeline

```
--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. Disabled 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. |

Lists of Reference Lamps Available

The automatic wavelength calibration relies on having previously calibrated reference lamps obtained in the same configuration or mode. It is also important that the lamp names are correct, for instance HgAr is quite different than HgArNe. For interactive wavelength calibration, reference lamps are used as a visual aid only. It lets you find the matching laboratory lines values that will be used to fit a pixel to wavelength relation that we call *Wavelength Solution*. The list of lamps is the following.

| Grating | Mode | Filter | Lamp |
|---------|------|--------|--------|
| 400 | M1 | None | HgAr |
| 400 | M1 | None | HgArNe |
| 400 | M2 | GG455 | HgAr |
| 400 | M2 | GG455 | HgArNe |
| 600-old | Blue | None | HgAr |
| 600-old | Blue | None | CuHeAr |
| 1200 | M2 | None | CuHeAr |
| 1200 | M3 | None | CuHeAr |
| 1200 | M5 | GG455 | CuHeAr |

Adding new reference lamps

It is possible to add new lamps very easily you just need a raw lamp that meets the following specifications with respect to your science project:

- Same instrument configuration or mode
- Same Grating
- Same order blocking filter if present
- Same binning
- Same lamp/combination that you use in your observations
- Smallest slit possible. Equal is OK too.

Then you can use the interactive mode or other software (such as IRAF) to produce a wavelength-calibrated 1D spectrum. Now you have to options, identify the system folder where the lamps that come with the package are saved and simply put it there or put it in another directory and use the argument --reference-files

```
redspec --reference-files /path/to/ref-lamp-location
```

Or contact storres [at] ctio [dot] noao [dot] edu and we will make it available as a package file.

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.

Note

IRAF is available in all three data servers. Running `iraf` will open an `xterm` and `ds9` windows. `iraf-only` will not open `ds9`

Establish a VNC connection

Separately, you should receive a server hostname, IP, port and VNC-password. If you don't you can ask for it. We have decided to use a similar organization of vnc desktops:

VNC ports and working folder assigned to each partner.

| Port | Partner/Institution | Folder |
|------|---------------------|---------------------------|
| :1 | NOAO | /home/goodman/data/NOAO |
| :2 | Brazil | /home/goodman/data/BRAZIL |
| :3 | UNC | /home/goodman/data/UNC |
| :4 | MSU | /home/goodman/data/MSU |
| :5 | Chile | /home/goodman/data/CHILE |

For the rest of this tutorial we will assume your host name is `vnc-server` the port 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`.

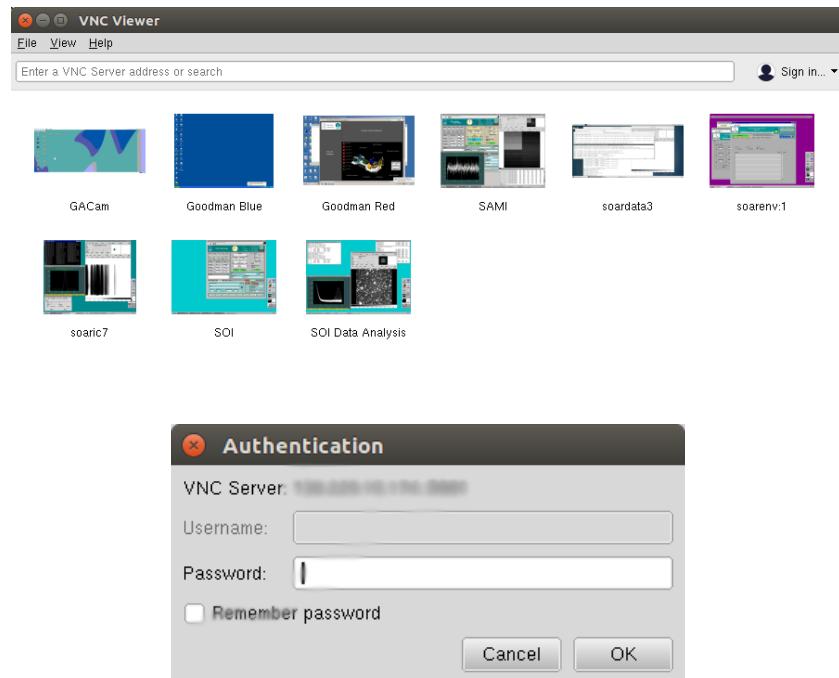
```
vncviewer vnc-server:1
```

You will be asked to type in the `password` provided.

VNC using a Graphical Client

Using a graphical VNC client is quite similar and intuitive

Running the Pipeline



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.

The terminal window title is "goodman@soardata3:~/data". The window contains a command-line session:

```
goodman@soardata3:~/data
File Edit View Search Terminal Help
[goodman@soardata3 ~]$ pwd
/home/goodman
[goodman@soardata3 ~]$ cd data
[goodman@soardata3 data]$ ls
BRAZIL CHILE MSU NOAO test UNC
[goodman@soardata3 data]$ df -h
Filesystem      Size   Used  Avail Use% Mounted on
/dev/nvme0n1p1    24G   11G   12G  49% /
devtmpfs        16G     0    16G  0% /dev
tmpfs          16G  172K   16G  1% /dev/shm
tmpfs          16G   18M   16G  1% /run
tmpfs          16G     0    16G  0% /sys/fs/cgroup
/dev/nvme0n1p2   211G   61M  201G  1% /home_local
/dev/md127       5.5T   42G  5.4T  1% /home
tmpfs          3.2G  20K  3.2G  1% /run/user/8143
tmpfs          3.2G     0  3.2G  0% /run/user/8142
[goodman@soardata3 data]$
```

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/ ./
```

Running the Pipeline

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:

If you are running `redccd` for the first time you can use `redccd` alone but if it's a second or third time you will need to use `--auto-clean` which is a built-in protection for your data, in case you don't want to delete what has been done. 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 --auto-clean
```

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

```
redccd --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 and it varies from one instrument configuration to another.

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
```

The meaning of every letter in `cfzsto` is summarized in the following table:

| Letter | Meaning |
|--------|--|
| c | Cosmic ray cleaned or mask created depending on the method |
| f | Flat corrected |
| z | Zero or Bias corrected |
| s | Slit trimmed, trims off the non-illuminated sections of the detector |
| t | Initial Image trimming |
| o | 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 we 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.

The final image has a `g` added to the start of the name, following the above example your final 1D and wavelength calibrated image will be named:

```
gcfzsto_image_file.fits
```

Using the Interactive Mode

If you need to make sure that your solution is the best possible you can use the *Interactive Mode*. Some of the key features are:

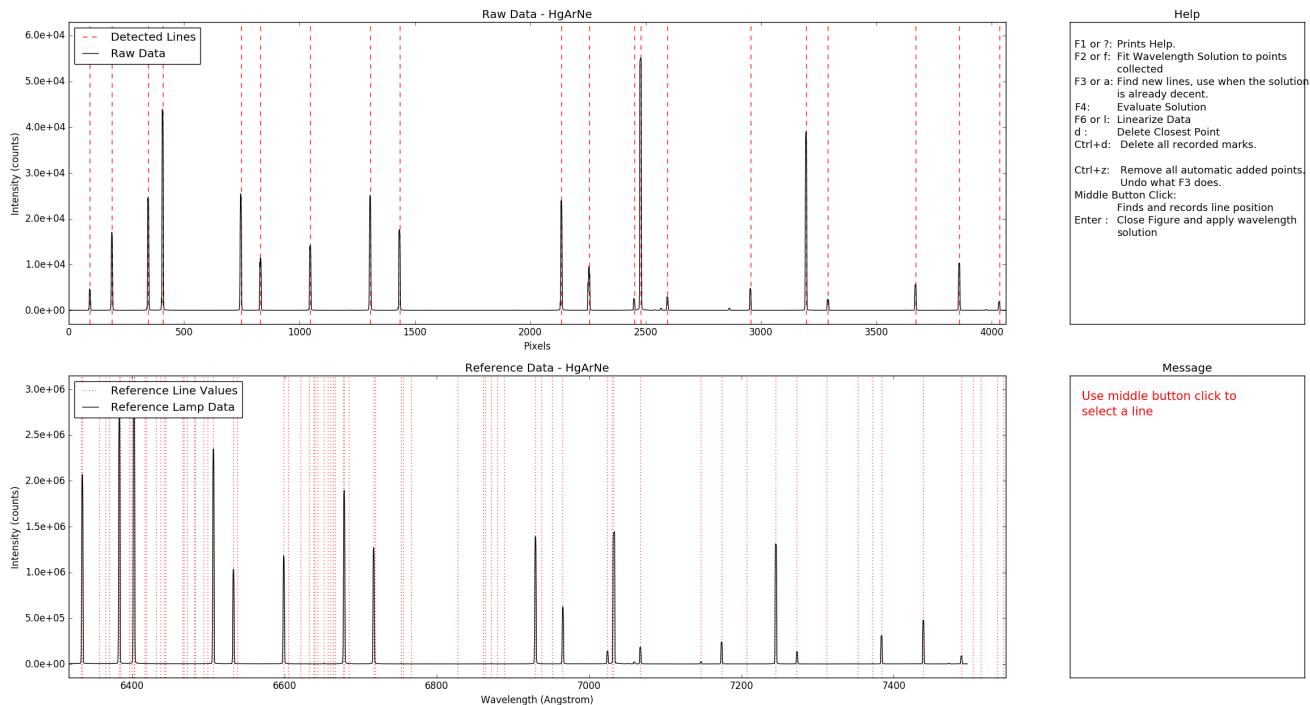
- Manually match spectroscopic lines
- Zoom in to get a better reference
- Evaluate the quality of your solution
- Uses a visual reference for matching the lines
- Uses laboratory values for reference lines

Right now it uses matplotlib as the underlying tool to enable interaction.

Below you will see a sequence as well as a description of the procedure to use the interactive mode.

1. The interactive screen has four subplots. In the *Upper main panel* you have the raw comparison lamp, i.e. your comparison lamp 1-D-extracted but without wavelength solution, the dashed red lines represent the lines found by the pipeline. In the *Lower main panel* you have the reference lamp. This is a previously calibrated lamp that is distributed with the package (also you can use your own). In this case the red lines are the laboratory values obtained from the NIST Atomic Spectra Database site (https://physics.nist.gov/PhysRefData/ASD/lines_form.html). The *Upper right panel* is a static help, intended to give you a quick and easy-to-reach help. Finally, the *Lower right panel* is an information window, in fact it serves three main purposes:

- Display messages and warnings.
- Shows you a zoomed line.
- Displays the wavelength solution quality information.



2. The way you interact is by using your mouse and keyboard. The following table describes all the available keyboard commands with their respective keys and/or buttons, and a summary of their functionality.

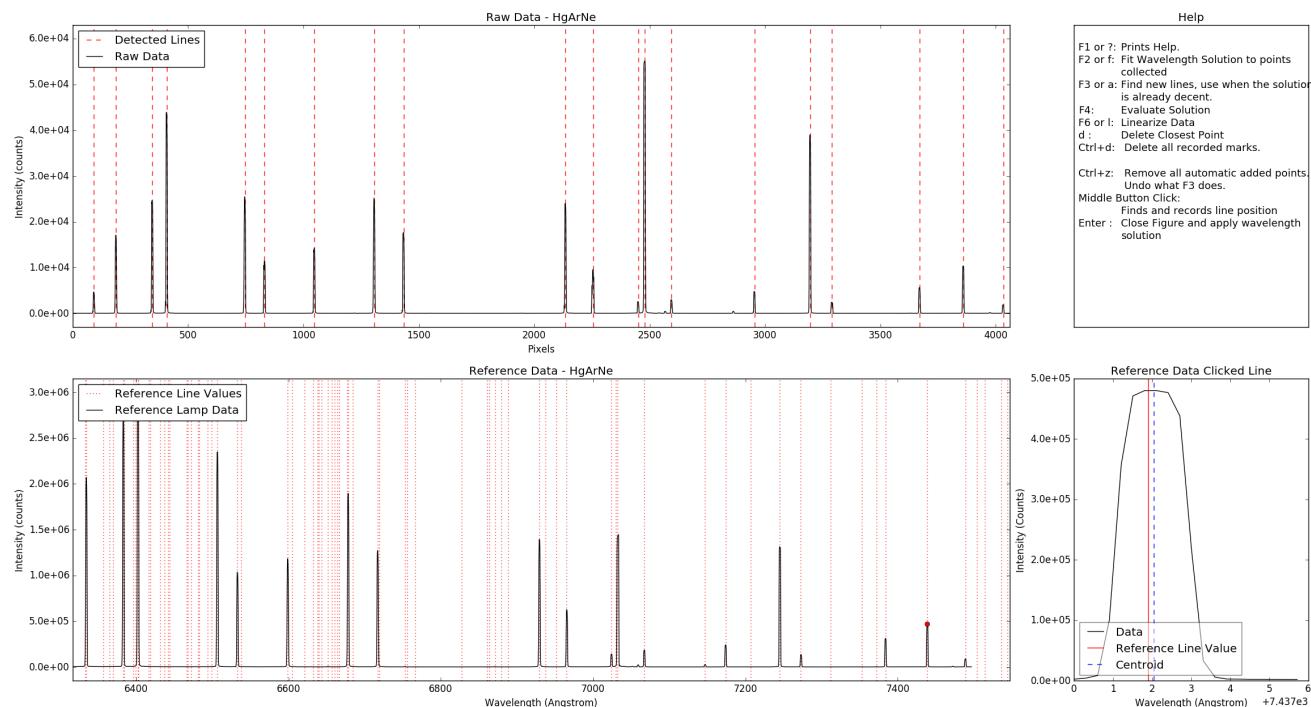
| Main Key | Alternative Key | Mouse Equivalent | Description |
|----------|-----------------|------------------|------------------------------------|
| ? | F1 | None | Print Help message on the terminal |
| f | F2 | None | Fit a function to collected points |
| a | F3 | None | Find lines automatically |

Running the Pipeline

| | | | |
|--------|------|---------------|---|
| None | F4 | None | Evaluate wavelength solution |
| d | F5 | None | Remove point closest to the mouse pointer |
| I | F6 | None | Linearize and smooth spectrum |
| m | None | Middle Button | Register the line closest to the mouse pointer |
| ctrl+z | None | None | Deletes ALL automatically added points |
| ctrl+d | None | None | Deletes all recorded points |
| ctrl+q | None | None | Ends the program |
| Enter | None | None | Accepts wavelength solution and closes the figure |

It is advisable to practice a little bit to get familiar with this combination of keys and functions. Since we are still in the development stage, if you feel that the use of a key creates problems for you let us know.

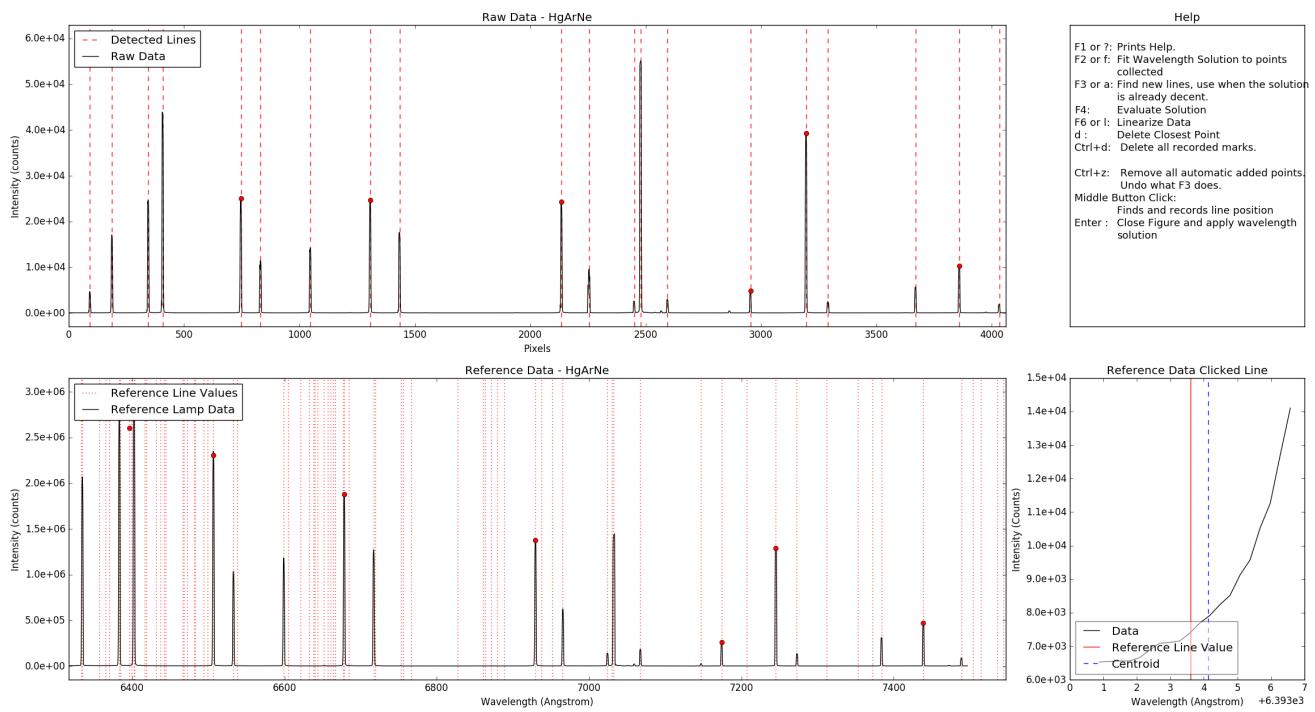
In the following image the mouse pointer was placed very close to the center of the line and then with a middle button click the line is marked with a red circle. The software will calculate a centroid (vertical dashed blue line) and then will try to find the closest reference value (vertical red line). Keep in mind that the reference data (red dotted lines) are not the *reference lamp* lines themselves but values obtained from the NIST site indicated above.



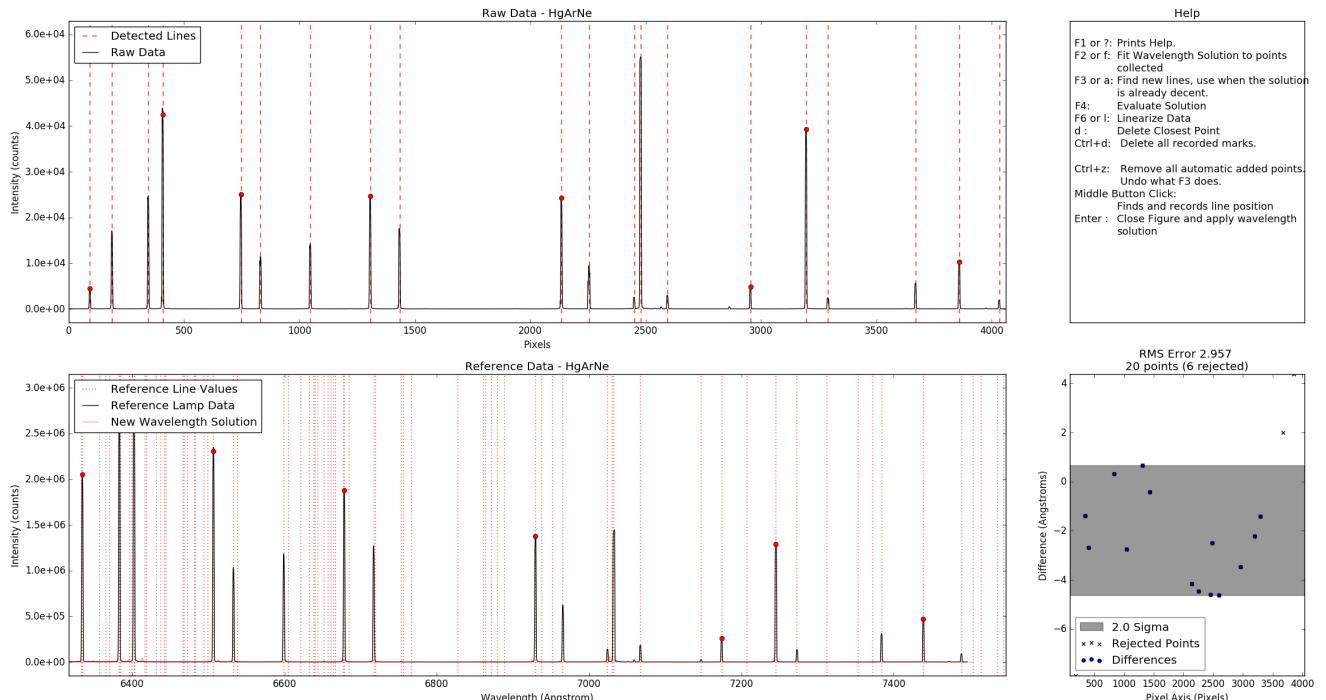
Then you have to find the corresponding line in the opposite plot. There is no preferred plot to start as long as you are consistent with the order in which you mark the lines.

3. In case you miss-identified a line, like in the image below, you can place the mouse pointer above the corresponding red circle and press **d** to delete it. It will search for the closest mark along the horizontal axis and remove it. If there is a counterpart in the opposite plot it will delete it too.

Running the Pipeline



4. Once you matched a good number of lines, the minimum required by the fitting routines are four, you can either press F2 or **f** to make a fit of the pixels and angstrom values collected. Now the *Lower Right panel* will show the scatter plot of the fit. It is important to note here that the points in this plot do not represent the points you marked but rather each of the lines detected in your extracted 1-D comparison lamp spectrum (red dashed lines in the *Upper main plot*) It does one iteration of a 2-sigma clipping to reject outliers, and then it uses those values to calculate the Root Mean Square Error. In the example we show here the RMS is a bit high, but we will fix that below.

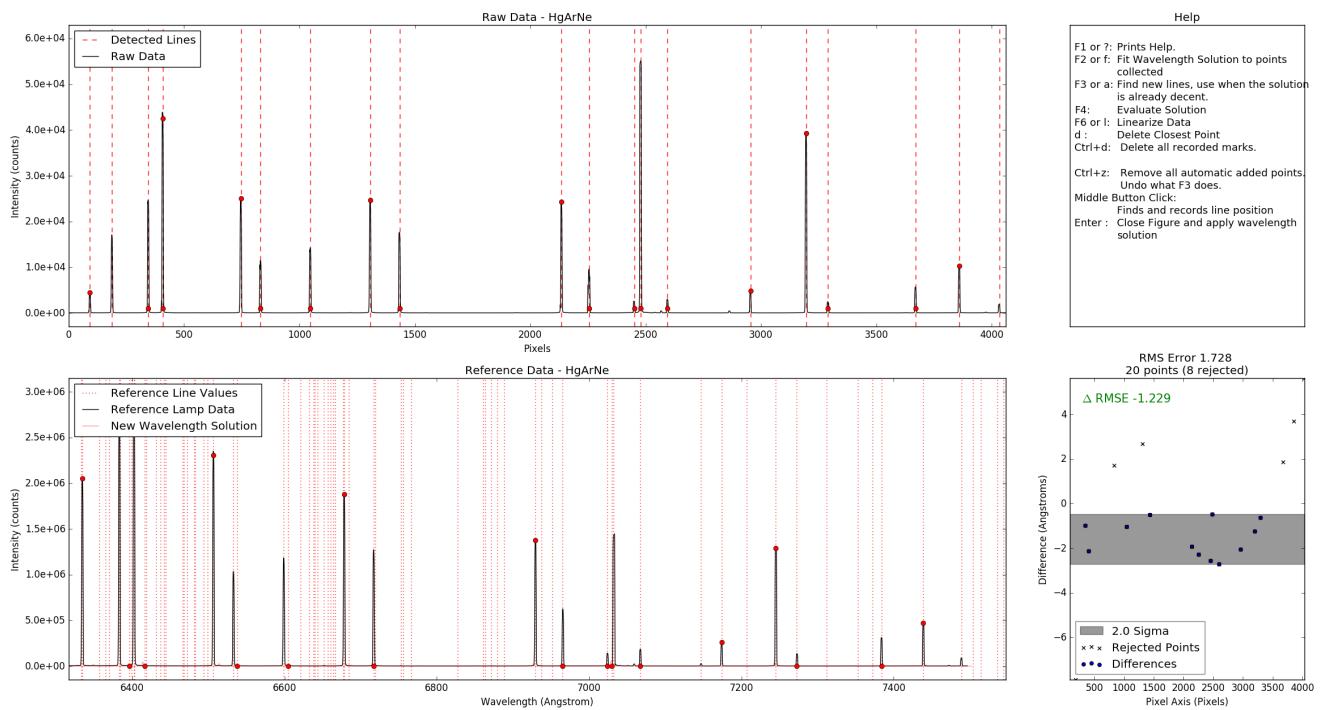


5. If you see that your current solution is decent you can press F3 or **a** and the pipeline will try to find more points automatically. The new matches found

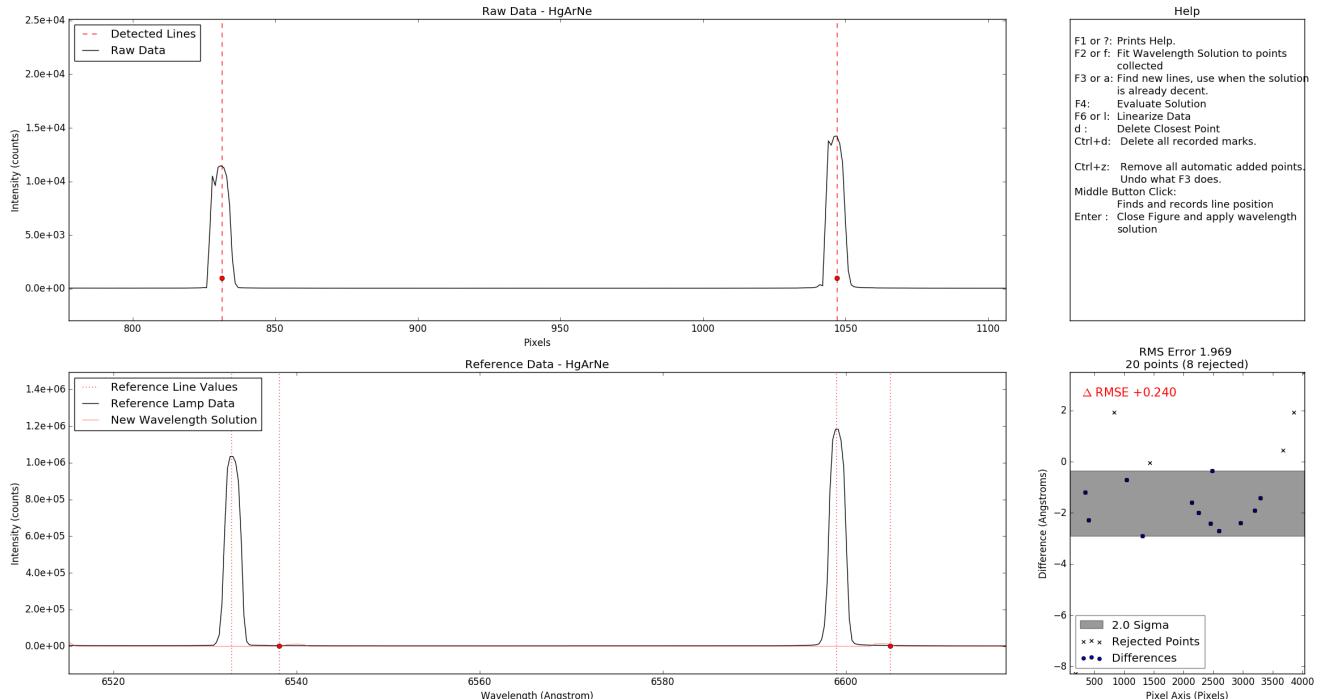
by the software are shown as red dots at the base of each line in the two main *Upper* and *Lower* panels.

The automatic finding routine is not perfect, and indeed it depends on the preliminary wavelength solution. It uses the detected lines in the uncalibrated 1-D lamp, applies the preliminary solution and tries to find a match in the reference line values. In most cases it improves the solution, but not always so keep that in mind. In this case the RMS error is reduced by almost half, which is good, but if you look closely you can see the mismatches; also the *Bottom right* panel will show you this.

Running the Pipeline

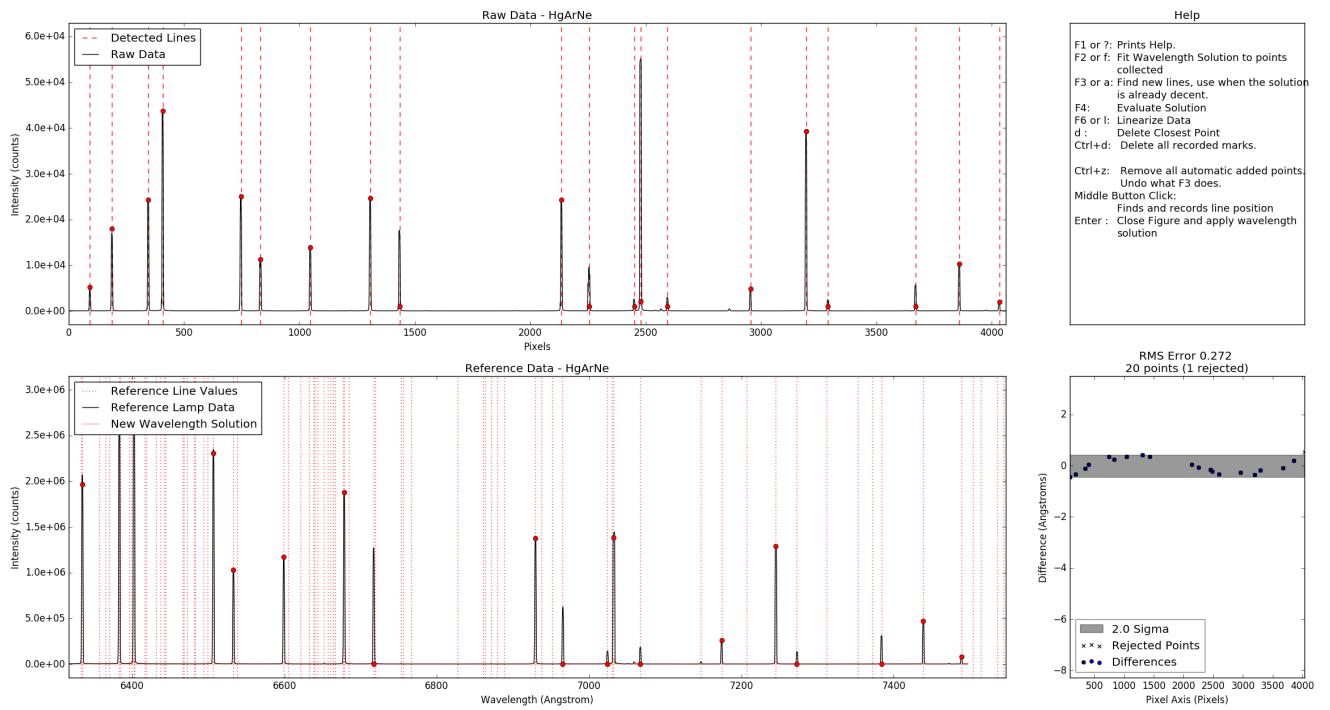


6. Now you we show an example of when the match is apparently good but in fact it's not. Here you need to zoom-in to see that there is an offset between the line centers and the matched laboratory line values, as shown in the figure below. You may have to apply different zoom values to your lamp and the reference lamp to get the plot to look like show it here. The solution in this case is locate the offending line(s), and delete it(the) pressing **d**. Then do a new match by clicking with the middle mouse button on the lines in the laboratory/reference plot, and the respective line in your 1-D uncalibrated lamp.



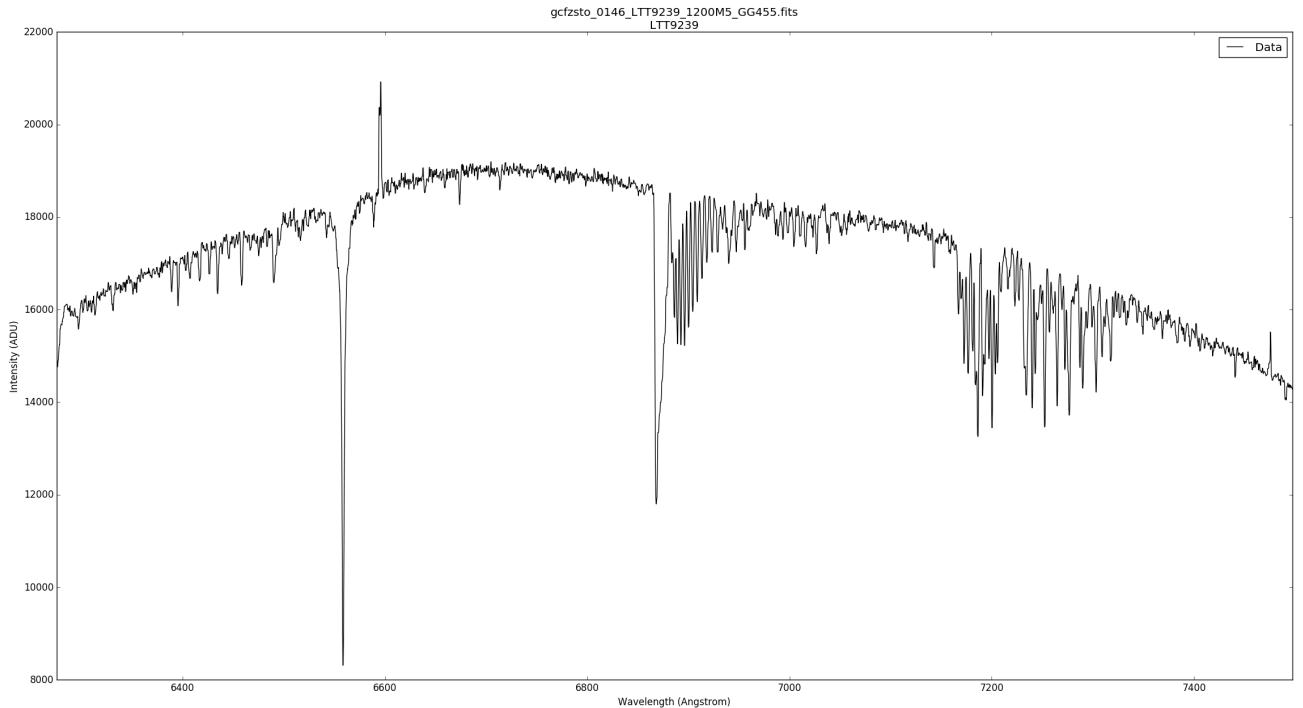
7. After you checked all the identifications and are happy with it, fit the solution again and you will obtain something like this:

Running the Pipeline

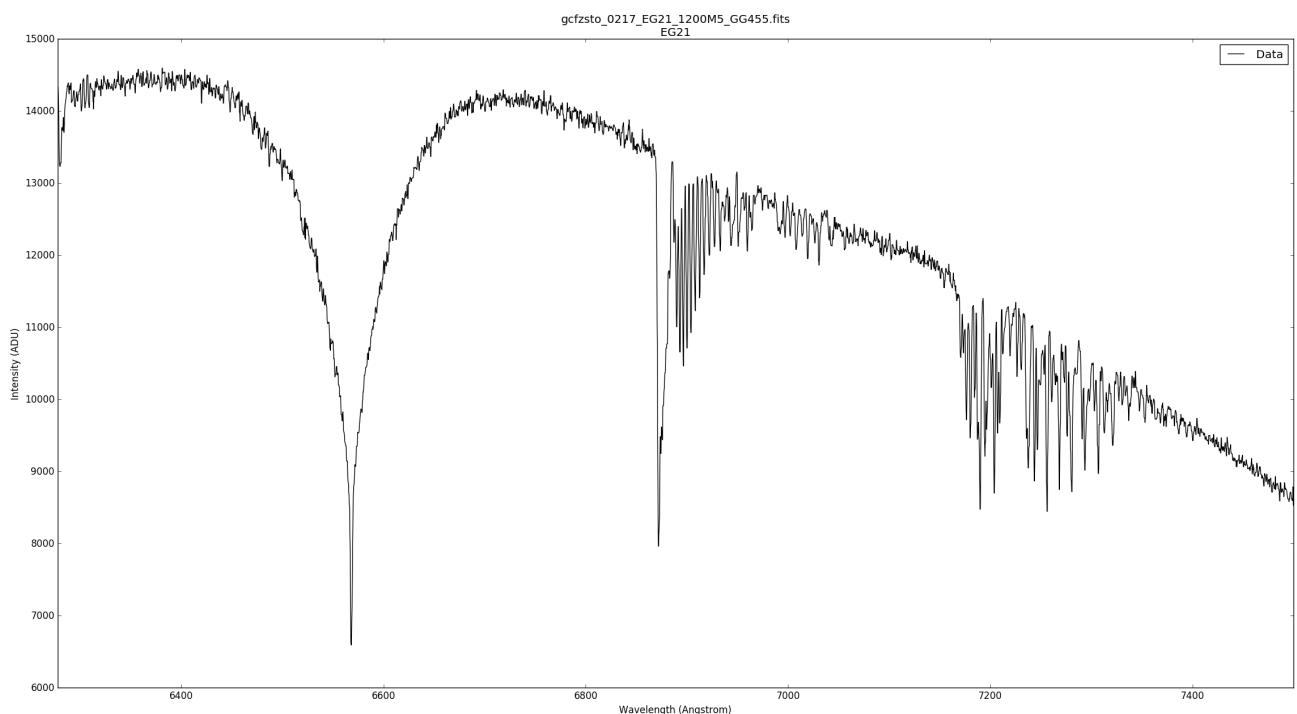
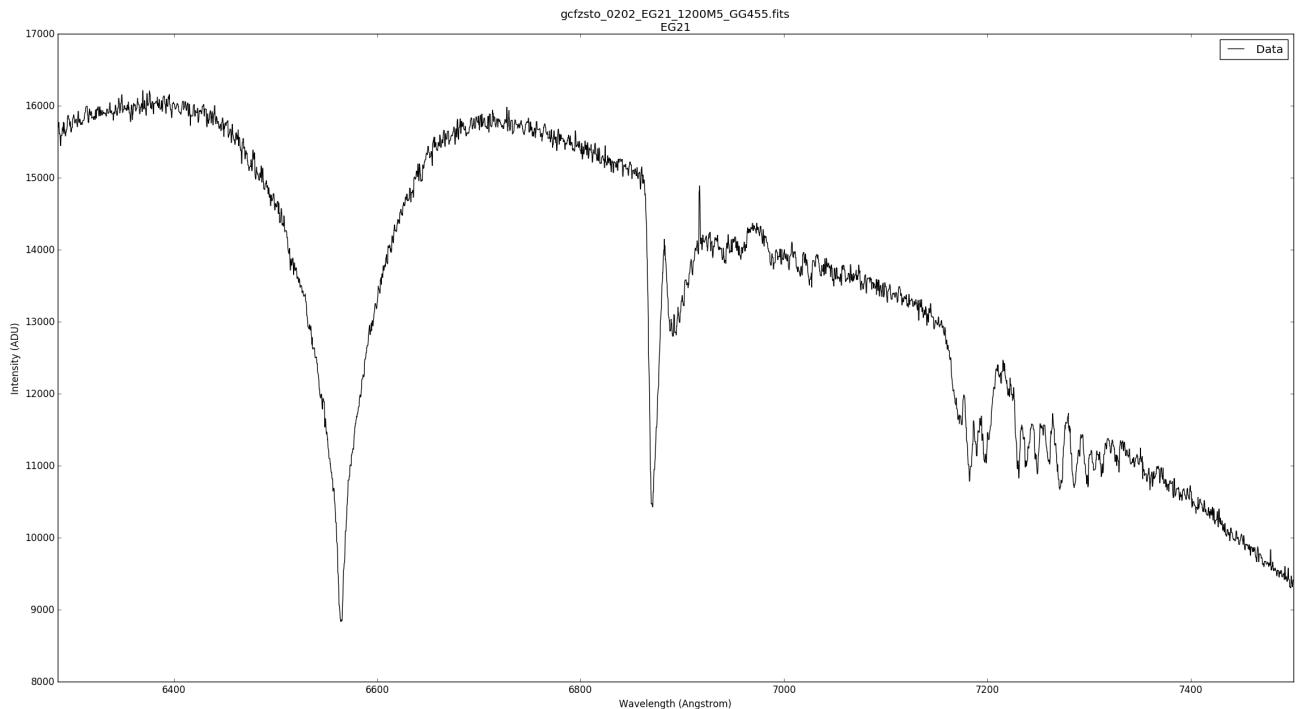


Here you can see that the *Bottom Right* panel shows the differences have a sinusoidal shape, which is also a sign that the solution can be improved. There are ways this can be implemented to refine the fit even further, but this is at present deferred to a later version.

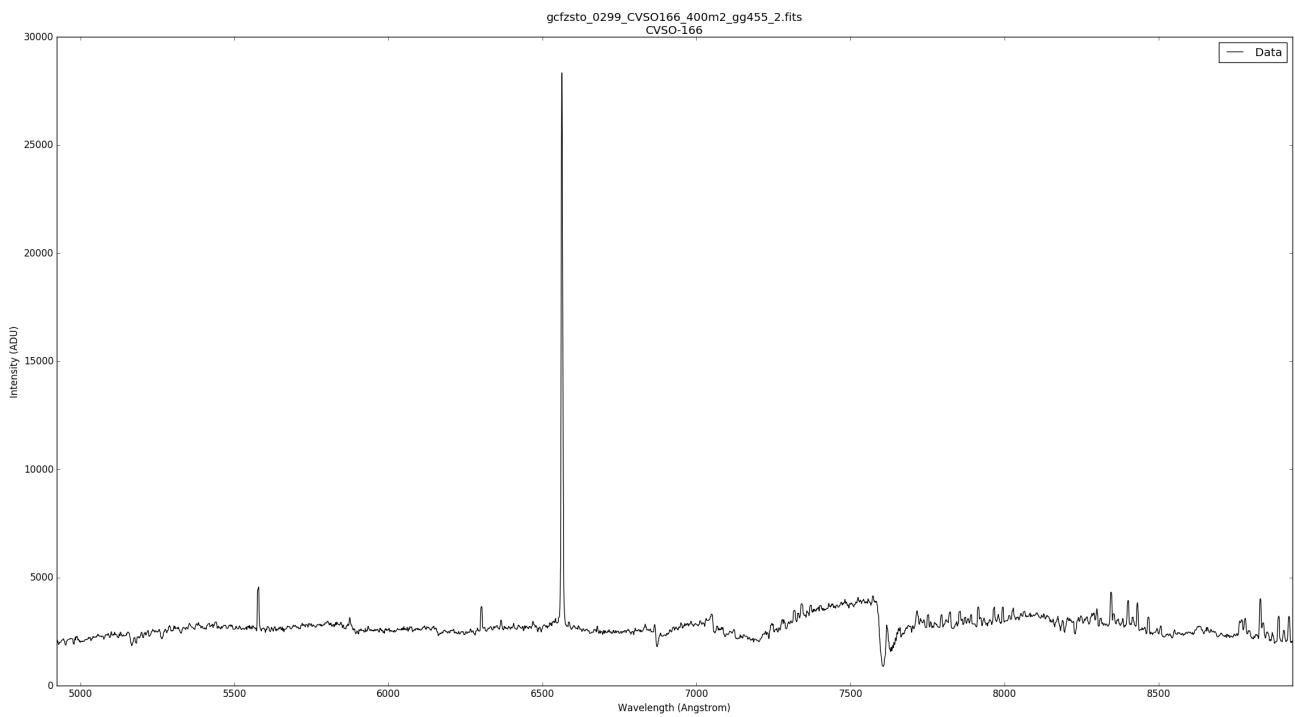
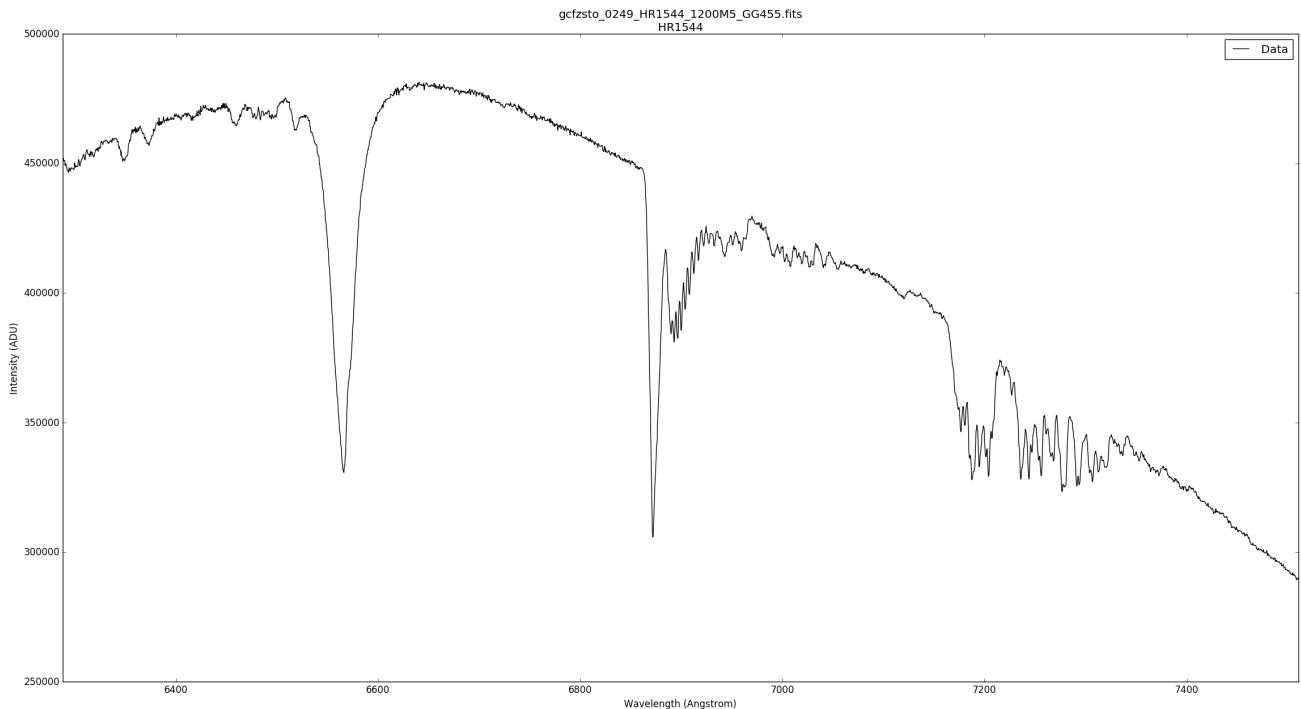
8. Finally, a few samples of the spectra extracted by the pipeline.



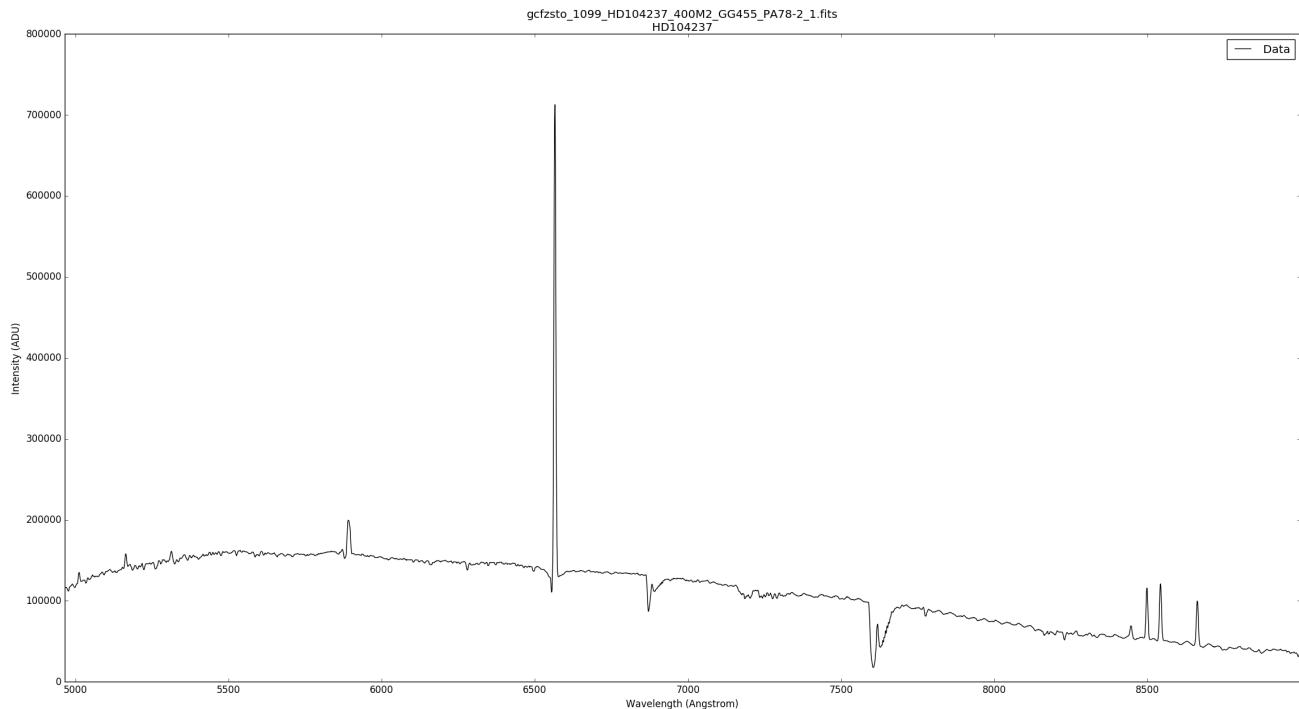
Running the Pipeline



Running the Pipeline



Troubleshooting



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 `OBSSTYPE` is correct.
- The reference data plot, in interactive mode, doesn't show anything or only vertical dotted lines: The reference lamp doesn't exist for that configuration, since this is used only for visual reference sometimes it will display the same lamp but in other instrument configuration, this will not affect the quality of the solution.

Installation Instructions

We strongly recommend installing the pipeline using *virtual environments*. Below you will find a summary of installation steps.

Warning

Remember that we are not providing any kind of support for installation. This documentation will be the only existing.

- Install anaconda
- Add astroconda channel
- Create virtual environment
- Activate environment
- Install requirements
- Install pipeline

Anaconda

For anaconda installation we recommend you to check the [astroconda channel's documentation page](#). The instructions will be reproduced here but they might change for newer versions.

Warning

Anaconda installer requieres BASH. Don't try with other shell.

1. Installing anaconda - Go to <https://www.anaconda.com/downloads> and download the appropriate *anaconda installer* for your platform, most likely it has been automatically selected.
 - Run the installer.

```
cd <download_directory>
bash <install_script>
```
 - Once completed, check the bottom of `~/.bash_profile` or `~/.bashrc` there should be a new `PATH` definition with anaconda included.
2. Check anaconda installation

```
which conda
```

You should get a response similar to this:
`~/bin/anaconda3/bin/conda`If you don't get this response check the detailed instructions on the astroconda site. Otherwise continue to the next step.
3. Configure Conda to use the *Astroconda Channel*

```
conda config --add channels http://ssb.stsci.edu/astroconda
```
4. Create a virtual environment. Here you have two options, one with `iraf` and one without it.
 - Standard: Without Iraf (Python 2 or 3).

```
conda create -n astroconda python=2.7 stsci
or
```

Install DCR

```
conda create -n astroconda python=3 stsci
```

astroconda is the name of your environment, you can use any name you want.

- Legacy software stack: Iraf included (Requires Python 2.7).

```
conda create -n astroconda python=2.7 iraf-all pyraf-all stsci
```

5. Activate your environment.

```
source activate astroconda
```

6. Get latest release of the *Goodman Spectroscopic Pipeline*

visit <https://github.com/soar-telescope/goodman/releases/latest> and download the *.zip or *.tar.gz

```
cd <download_location>
```

```
tar -xvf <pipeline_file>.tar.gz
```

or

```
unzip <pipeline_file>.zip
```

7. Install requirements from requirements.txt

```
cd <goodman_pipeline_unpacked_location>
```

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

8. Install the pipeline

```
pip install .
```

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 away. 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 likely 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.

Install DCR

5. Close and reopen the terminal or load the .bashrc file.

```
source ~/ .bashrc
```