User Manual for Data Reduction of the EXOhSPEC Instrument Working version: V0.0.1

Neil Cook, Ronny Errmann
Physics, Astronomy and Maths, University of Hertfordshire

January, 2018

Contents

1 Introduction

This program is designed to take a fits image file (i.e. a CCD image) and extract out orders from an Echelle spectrograph (regardless of separation and curvature,

as long as orders are distinguishable from one-another).

2 Installation and dependencies

This program was written to work with modules installed in Anaconda 2 and python 2.7. It is recommended to use an installation of anaconda 2 to use this program (If this is not possible, the whole module dependency is given in Appendix A).

2.1 With Anaconda and python 2.7

Anaconda can be downloaded from https://www.continuum.io/downloads. Please select the version for python 2.7. After download it can be installed by running

```
sh Downloads/Anaconda2-5.0.1-Linux-x86_64.sh
```

(no superuser permissions are needed).

After the installation and opening a new terminal python should point to the one in the Anaconda installation. It can be checked by running `which python` . If the result doesn't point to your Anaconda installation, but somewhere else (e.g. `/usr/bin/python`), an extra entry into `.bashrc` or `.tcshrc` needs to be added. If you use `bash` add the following line (replace */home/exohspec* by the path you used for the installation to your `.bashrc`.

```
export PATH="/home/exohspec/anaconda2/bin:$PATH"
```

If you use `csh` add the following line (replace */home/exohspec* by the path you used for the installation to your `.tcshrc`

```
setenv PATH /home/exohspec/anaconda2/bin\:$PATH
```

After making the change and opening a new terminal python should point to the file in the Anaconda installation (`which python`).

2.1.1 Additional modules

Once Anaconda is installed one other module is needed (tqdm). If Anaconda is installed correctly then pip can be used to install this module

```
pip install tqdm
```

2.2 Necessary python files

The following files need to be located in the program folder:

create_badpx_mask.py Script to create a bad pixel mask.

reduction_day.py Script which needs to be run on each new set of data. It creates the data, which is necessary in order to extract the scientific spectra.

extract.py Script that extracts the scientific echelle spectra.

order_trace.py Script that searches for the flat orders in the CCD images.

procedures.py Contains all the procedures for the data analysis.

tkcanvas.py Contains the plotting routines to create the user interfaces (UI).

plot_img_spec.py This script controls plotting of CCD images, graphs, and the extracted spectra.

remove_orders.py This needs to be run in order to find an initial wavelength solution (will be included in reduction_day.py later).

When running the scripts for the first time, python will create a new file called `<filename>.pyc` for some of the files. No harm will be done in removing or keeping the files.

2.3 General information about the way the pipeline works

Handling parameters

1. The script has some hard coded values in the *.py files (normally at the beginning of a procedure). These values can be changed for testing, but usually do not need any changes.
2. The pipeline reads the parameters from a configuration file (standard: `conf.txt`). These parameters need adjustment on a regular basis (see Section 3.2).
3. The parameters which are set in the configuration file can be overwritten with a command line input when a python script is started (e.g. `python bla.py argument1=value1 argument2=value2`).
4. Some parameters can be overwritten during the run time of the script by user input.

Finding further information

- The pipeline logs the execution of procedures and necessary information in a logfile (standard: `logfile`). This information is also printed into the terminal window.
- All adjustable parameters are logged at the beginning and at the end of the execution of a python script into a logfile (standard: `logfile_params`).
- Images with the results of some steps can be found in a subfolder (standard: `logging`). Some of these images are shown in Figures 1, 2, 3, or 4.
- Each parameter is explained in detail in the configuration file.
- The input and output parameters of the individual procedures are explained in the file `procedures.py`.

2.4 Short introduction to reduction of CCD images

The data stored in CCD images is affected by the physical parameters of the individual pixels and way the readout electronics work. Therefore each scientific frame needs to be corrected (pixel by pixel) with the formula:

$$\text{reduced Science} = \frac{\text{raw Science} - \text{master Bias} - \text{master Dark}}{\text{master Flat}}\,. \tag{1}$$

The master Dark should have the same exposure time as the Science frame and the master Flat should be normalised in order to keep the flux levels. To create the master Dark and master Flat, the following steps are necessary:

$$\text{master Dark} = \text{raw Dark} - \text{master Bias} \tag{2}$$

$$\text{master Flat} = \text{raw Flat} - \text{master Bias} - \text{master Dark}\,. \tag{3}$$

The master Dark used for the creation of the master Flat should have the same exposure time as the Flat and needs to be corrected with the Bias. The master Bias in each of the formulas can be different.

In order to decrease the noise levels, each of the master file should be created by median combining several (corrected) images together. If the brightness of the individual flats vary, then the flats should be weighted by their median flux before combining them. Combining all steps leads to the formula

$$\text{reduced Science} = \frac{\text{raw Science} - B_S - (D_S - B_{D_S})}{F - B_F - (D_F - B_{D_F})}\,, \tag{4}$$

where B_S is the master Bias, created by combining biases taken at a similar time as the science frame, D_S is the master Dark, created of darks with the same exposure time as the science frame. In case the bias level and noise don't change during the night, then Equation 4 changes to

$$\text{reduced Science} = \frac{\text{raw Science} - D_S}{F - D_F}\,. \tag{5}$$

In case the dark level is negligible and the Bias level is constant, then Equation 4 changes to

$$\text{reduced Science} = \frac{\text{raw Science} - B}{F - B}\,. \tag{6}$$

In the pipeline always the individual frames are corrected before frames are combined.

3 Running the code on a new set of data (e.g. a new night)

The following steps will be performed on a new set of data:

1. Creating the following reduced and combined files:

   sflat File in which the science traces can be determined (standard: 5x white light flats, best without arc lamp). This file is also used in order to create a map of the background flux

   arc File in which the wavelength calibration traces can be determined (standard: 5x ThAr alone (future development: white light flats)).

   arc_l : Long arc exposure to create a good wavelength solution over the whole chip (standard: 5x 60 s ThAr).

   arc_s : Short exposure in order to find the center of the saturated lines in arc_l (standard: 5x 10 s ThAr taken between the arc_l images).

   flatarc : File which contains the flat in the science fiber and the arc in calibration fiber (standard: 5x files with the white light flats in science fiber and ThAr in the calibration fiber).

2. Determining the shift of the science traces compared to the previous solution (e.g. previous day). If the instrument was not touched, the shift should be small and constant (→ Fig. 1). If the file for the previous solution does not exist, or if a big deviation to the previous solution has been found, then another step will be executed:

   2a. Finding the traces of all science orders.

3. Creating a map of the leaking light between the science orders (background map).

4. Finding the traces of the calibration orders (→ Fig. 2).

5. Create the wavelength solution for the night (→ Fig. 3 and 4). This is based on the solution of the previous data set, which is adjusted according to the given parameters.

6. Extract the flat and normalise it.

If the result file of any of the above steps already exist in the folder in which the code is run, then the existing file is read instead of performing the step again.

3.1 Necessary CCD images

To get the best results, the following data should be taken:

- *true Flat (at least 11 files of the evenly illuminated CCD)*
- Flat (3x, white light source through the science fiber)
- Arc (5x, calibration lamp through the calibration fiber, alternating a short and a long exposure time)

- FlatArc (11x, white light source and calibration lamp)
- Bias (11x) and/or Darks (11x, for the exposure time of the true Flat and FlatArc)

3.2 Steps to run the code on a new set of data

1. Create and enter a new folder. This folder will be used for the reduced and extracted data.
2. Copy the configuration file (`conf.txt`) into the folder. The following parameters might need to be changed in the file `conf.txt`:

   GUI : If set to true, this allows manipulation of some parameters in a user interface (UI) during the runtime of the script.

   raw_data_path : Folder to the raw data.

   *_rawfiles] : Replace any filenames, which are different in the current observation

   ***_calibs_create** : Change the calibration steps, which are applied to the raw files before combining them. A list of possible calibrations are given in Section 4.

   original_master_order_filename : Path to the traces of the science orders from the previous day. This file will be used for [Step 2] as described in Section 3. Leave empty if the science orders should be searched without taking the previous solution into account (e.g. the setup of the spectrograph has changed).

   original_master_arc_solution_filename : Path to the previous wavelength solution. If this file does not exist, then the script will create a new solution by reading the file `arc_lines_wavelength.txt`. For a description of the format of this file, check Section 3.4.

3. Run the python script: `python <path to scripts>/reduction_day.py`

After checking the results (see Section 3.3), one might want to remove wrongly identified orders. This can be done in a graphical user interface by running `remove_orders.py`. The old files will be moved into a sub-folder and the necessary steps of `reduction_day.py` will run again.

3.3 Results from the pipeline

Step 1: The reduced and combined CCD images will be stored in the folder where the pipeline was run, if the parameter `master_<type>_filename` exists. Thereby `<type>` defines the image type, e.g. bias, sflat, or arc_l.
Step 2: The trace of each scientific order is stored in the file given in parameter `master_order_filename` (standard: `master_orders.fits`). This file contains one line for each order, normally starting with the reddest order, which is located on the left side of the CCD image. Each line contains the following information:

- Number of the order

- Parameters of a polynomial fit to the trace, e.g. 6 values if the trace is fitted with a polynom of order 6
- The lowest and highest pixel in dispersion axis, for which the trace can still be identified.
- The last three entries of each line define the width of the trace: the width from the center to the left, the width from the center to the right, and the Gaussian width.

Step 3: The background map, 2d image is stored in the file given in parameter `background_image_filename` (standard: `background.fits`). Additionally a mask which provides the pixel which are used to create the background map is stored in the file given in parameter `background_px_filename` (standard: `background_px.fits`).
Step 4: The traces of the orders of the calibration fiber are stored in the file given in parameter `master_orderarc_filename` (standard: `master_ordersarc.fits`). The format of the file is the same as for the scientific orders. The traces of the calibration orders are only shifted from the scientific orders while the curvature of the traces is kept the same.
Step 5: The wavelength solution is stored in the file given in parameter `master_arc_solution_filename` (standard: `master_arc_solution.fits`). The parameters for each order are stored in one line. For each order the following values are stored:

- Real order, as derived from the grating equation.
- Central pixel of the order.
- Parameters of a polynomial fit to the trace, e.g. 4 values if the dispersion axis is fitted with a polynom of order 4.
- List of the wavelengths of all the identified reference lines in this order.

During the step to find the wavelength solution, the pipeline logs data similar to the following output:

```
Info: To match the most lines in the arc with the old wavelength solution, a shift
of -2 orders, a multiplier to the resolution of 0.994, a shift of -10 px, and a
shift of 0.0 px per order needs to be applied. 1046 lines were identified. The
deviation is 0.1197 Angstrom.
Info: used 524 lines. The standard deviation of the fit is 0.0054 Angstrom. A 2d
polynom fit with 4 orders along the traces and 4 orders perpendicular to the orders
was used. With this solution, the offset to real orders is 72. With this offset, the
standard deviation of the residuals between the central wavelengths and the grating
equation is 0.1295 Angstrom. Using the original solution gives an offset of 72.
order   cenwave minwave maxwave ranwave Ang/px  name    number  gausswi gausswi min_     max_     range_reflines
                                                            dth_avg dth_std refline  efline   _whole_order

0       7978.6  7894.7  8054.2  159.5   0.038   Ar I    2       1.78    0.58    7916.4  7948.2   98.1
0       7978.6  7894.7  8054.2  159.5   0.038   Th I    2       2.04    1.15    7937.7  8014.5   98.1
...
27      5800.1  5769.1  5854.3  85.2    0.027   Ar I    2       3.06    0.15    5802.1  5834.3   44.6
27      5800.1  5769.1  5854.3  85.2    0.027   Th I    6       2.31    0.41    5789.6  5832.4   44.6
-1      -1.0    -1.0    -1.0    -1.0    -1.000  Ar I    148     2.07    0.61    5802.1  7948.2   2146.1
-1      -1.0    -1.0    -1.0    -1.0    -1.000  Th I    376     2.04    0.66    5789.6  8014.5   2224.9
```

Thereby the table contains the following information:

order Order of the trace, starting with 0 for the reddest order ($\tilde{m}$). To get the real order the offset m_0 is given in the text before (here: 72). The offset is determined using two different ways. First the data is compared to the grating equation $\lambda \propto m_0 + \tilde{m}$. In Practical this was done by searching for the smallest slope in the formula $y = (m_0 + \tilde{m})\lambda_c$, were λ_c is central

Figure 1: Reduced CCD image of a white light flat (log10 gray scale) with the marked traces of the identified scientific orders (red). The extraction width is given in dashed lines.

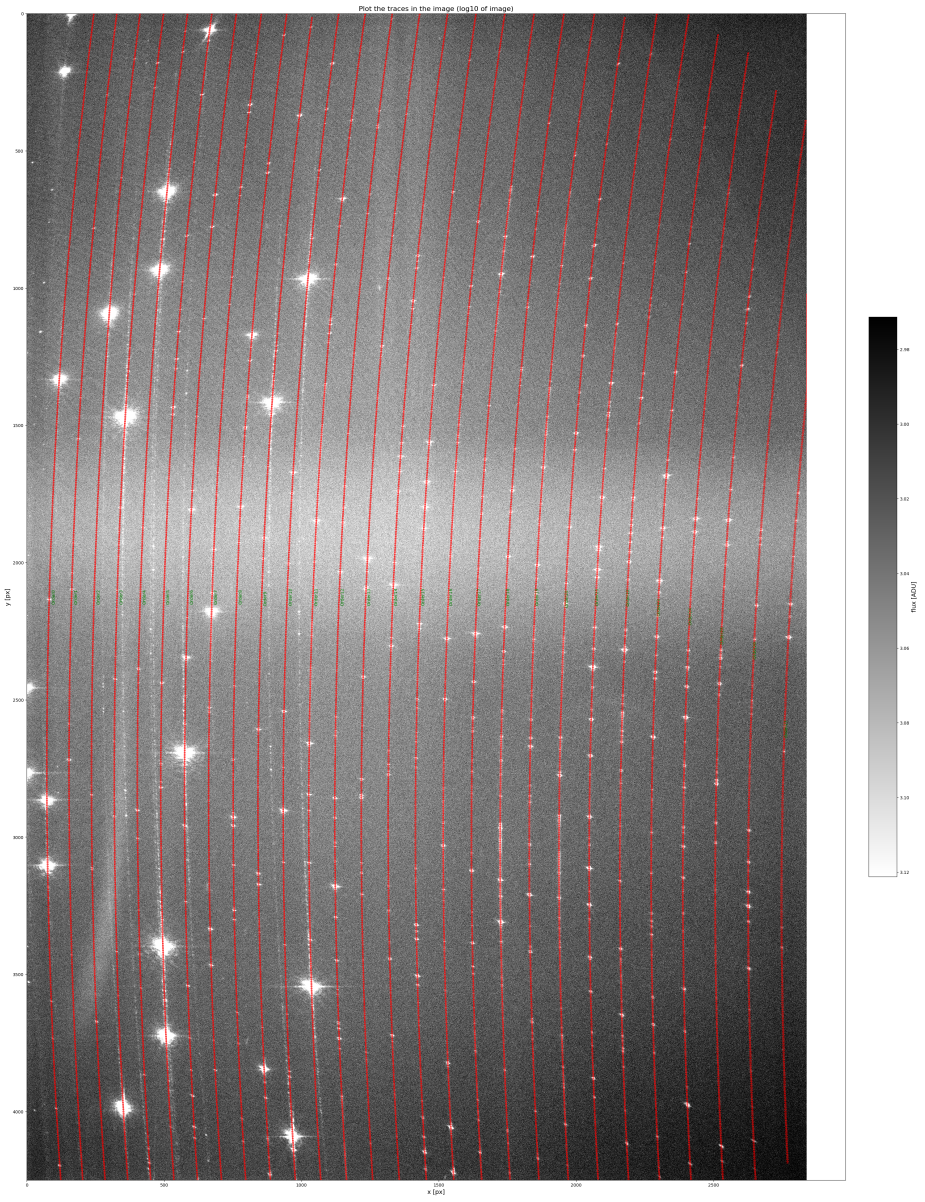

Figure 2: Reduced CCD image of a ThAr exposure (log10 gray scale) with the marked traces of the identified calibration orders (red). The extraction width is given in dashed lines.

wavelength of each order. The second value for the real order offset is determined from the shift towards the previous wavelength solution. Both values for the offset should be the same. Only the first value is saved in the file.

cenwave Central wavelength of the order, determines the zero point of the wavelength solution.

minwave, maxwave, ranwave Minimum and maximum wavelength, and wavelength range which is covered by the trace of this order.

Ang/px Resolution in $\frac{\mathring{A}}{\text{px}}$ at the central wavelength.

name Type of the reference line. If different types of reference lines were found in the order then a output for each available type is created.

number Number of reference line for this type and order.

gausswidth_avg, gausswidth_std Average and standard deviation of the Gaussian width of the emission lines

min_refline, max_refline Minumum and maximum wavelength of the reference lines of this type and order

range_reflines_whole_order Wavelength range which was covered by reference lines for this order (independent of type of the line)

The last lines give the information for all orders. Columns, for which no useful information can be derived show the value `-1`.
Step 6: The normalised white light flat is stored in the file given in parameter `master_flat_spec_norm_filename` (standard: `master_flat_spec_norm.fits`). The extraction and data format is described in Section 4

3.4 Creating a new wavelength solution

If no previous wavelength solution exist, then wavelengths and pixel need to correlated manually. In this case the parameter `original_master_arc_solution_filename` needs to point to a file, which does not exist (e.g. *master_wavelength_manual.fits*). Then the python scrip can be run normally. It will stop after searching for the lines in the emission line spectra. The order and pixel position of all identified lines are stored in the file given in parameter `logging_found_arc_lines`. This file can be opened and the corresponding wavelengths can be added to some of the lines. I have found that providing 1.5 lines per order is enough for the pipeline to do the rest. The lines should cover as much of the CCD image as possible.

The correlated data needs to be saved into file `arc_lines_wavelength.txt` with the following tab-separated entries:

- Order (starting at 0)
- Real order (bigger than 0, usually between 60 and 120)
- Pixel

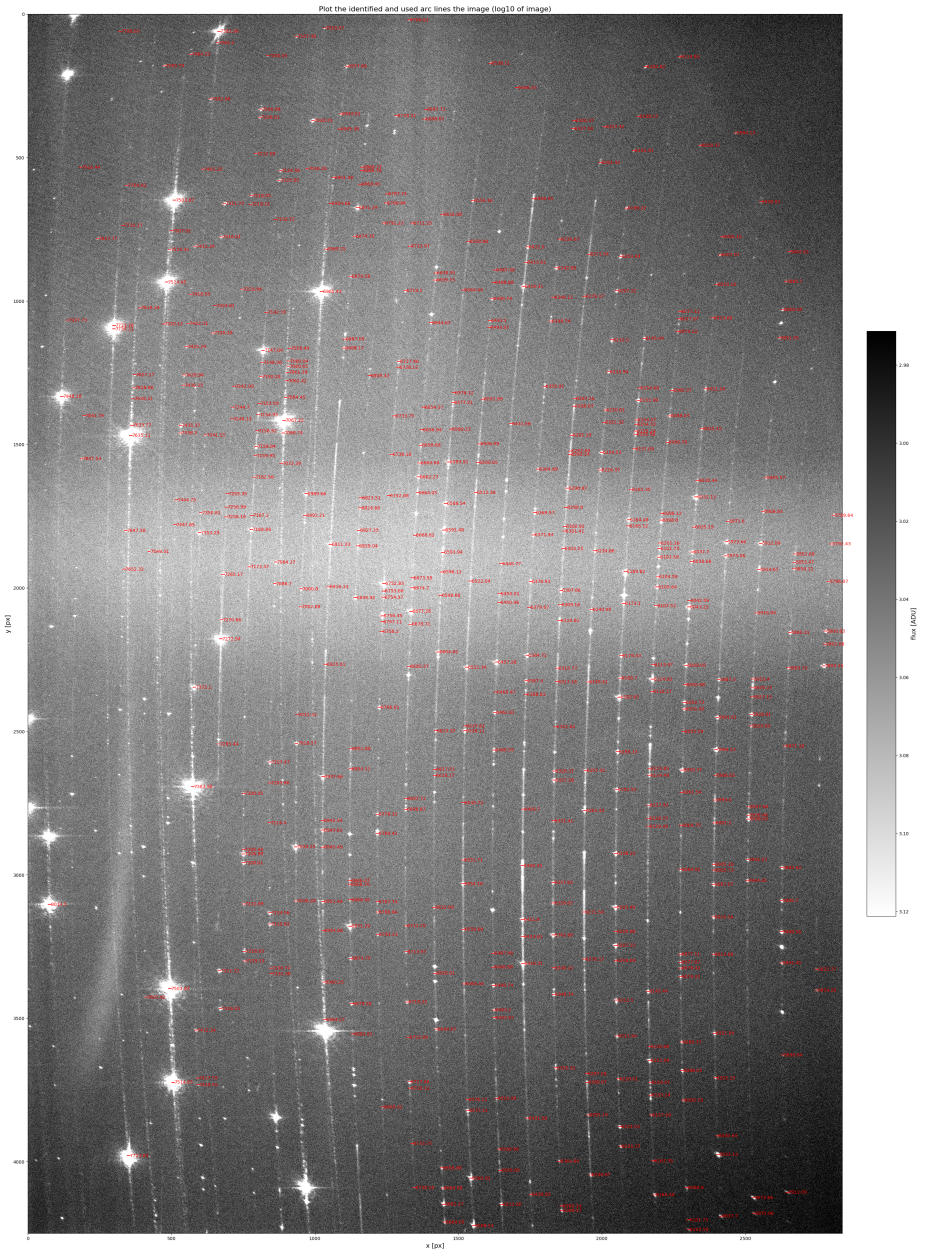

Figure 3: Reduced CCD image of a ThAr exposure (log10 gray scale) with the identified lines from the reference catalogue (red). Only this set of data was used to create the wavelength solution.

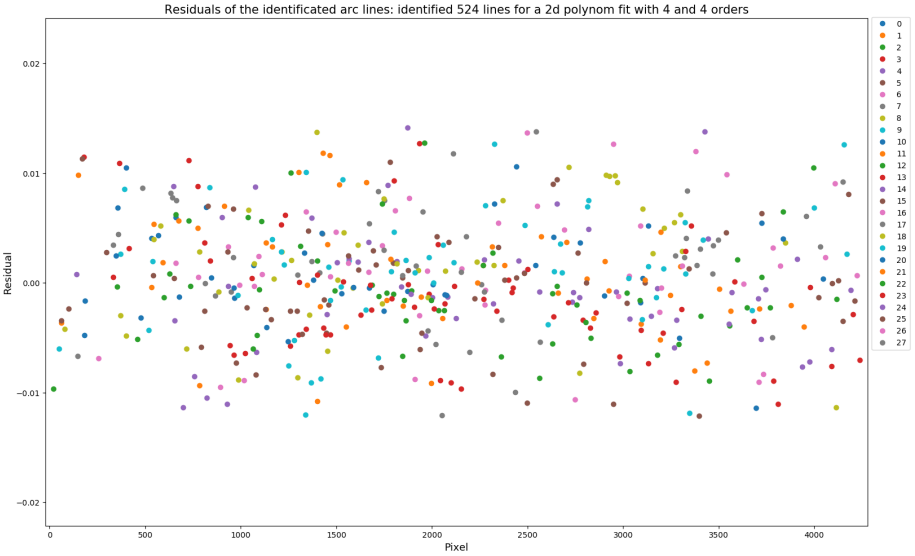

Figure 4: Residuals between the identified cataloge lines and the wavelength solution. Colorcoded are the different orders.

- Wavelength
- Type of the line (e.g. ThI, NeII).

Afterwards the script can run again and will use this data to create a new wavelength solution.

4 The Extraction of science data

Once the preparation steps have been run for a new set of data (see Section 3) the scientific data can be extracted. This is done with the python script `extract.py`.

Before running the script the name of the files which should be extracted needs to be given. afterwards the script can be run using

```
python <path to script>/extract.py
```

The script will load all necessary files (created by the step above, see Section 3) and afterwards perform the following steps:

1. Apply the corrections to the CCD image as given in parameter `calibs`. The following options are possible:

   subframe Use only an area of the CCD image.

   badpx_mask Apply the bad pixel mask, given in parameter `badpx_mask_filename`.

   bias Apply a master bias, which will be created from the files given in the parameter `bias_rawfiles` by using the settings given in the parameter `bias_calibs_create`.

   dark Apply a master dark, which will be created from the files given in the parameter `dark?.?_rawfiles` by using the settings given in the parameter `dark?.?_calibs_create`. The `?.?` stands for the exposure time of the science observation as found in its header. Alternatively, to force the use of a specific dark, the entry needs to consists of 'dark' with at least one additional character.

   flat Apply a master flat, which will be created from the files given in the parameter `flat_rawfiles` by using the settings given in the parameter `flat_calibs_create`.

   background* (standard: `background_image_filename`) Applies background correction using the filename given in the parameter `background*`. The background image will be scaled by the exposure time of the scientific image, before its subtracted from the scientific image.

2. Find the shift to the file containing the traces. Due to thermal movement the CCD might can move in respect to the instrument.

3. Extract the science fiber spectrum (and error).

4. Extract the calibration fiber spectrum.

5. Find the shift to the wavelength solution. The identified lines from the reference catalogue are searched and fitted in the calibration fiber spectrum. Then the shift is calculated between the original and fitted position.

6. Calculate the wavelength for each order and pixel by solving the shifted wavelength solution.

7. Create the flat normalised spectrum (and error).

8. Create the continuum corrected spectrum.

9. Perform some general measurements which are stored in the header.

10. Writing the file with all extracted data into the folder given in the parameter `path_extraction` (standard: `extracted`).

4.1 Format of the extracted files

The final fits file contains the original header and some additional information, for example the calibration steps which were applied and some information about the flux collected in the different orders. The data in the file is stored in a 3D array in the form: data type, order, and pixel. The data types are similar to the ones created by the CERES pipeline and are the following:

0. 2D array with the wavelength for each order and pixel.

1. Extracted spectrum without any modification.

2. (Measurement of the error the extracted spectrum)

3. Flat corrected spectrum, calculated by dividing the extracted spectrum and the normalised flat spectrum.

4. (Error the flat corrected spectrum)

5. Continuum normalised spectrum. The continuum is derived by fitting a polynomial to the flat corrected spectrum, using only areas of the spectrum where no lines are located.

6. Signal to noise ratio in the continuum, calculated from the residuals between continuum fit and measured continuum and the flux in the continuum.

7. Mask with good areas of the spectrum. The following values are used:

1	good
0	no data available
0.1	saturated pixel in the extracted spectrum
0.2	bad pixel in the extracted spectrum

8. Spectrum of the calibration fiber, e.g. of the emission line lamp.

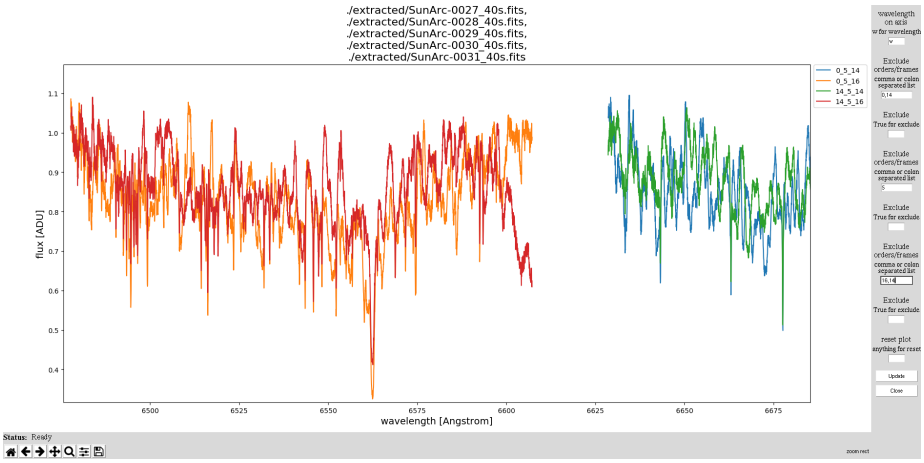

Figure 5: An example of how to plot the extracted data. Shown are the solar spectra of entry 0 and 14 in the plot_files.lst file. The graph shows the continuum normalised flux (data type 5), plotted over the wavelength (w). Only orders 14 and 16 are shown.

5 Post-extraction analysis

Once the science spectra were extracted these files can be analysed more in detail.

5.1 Plotting the data

To plot the results the python program `plot_img_spec.py` can be used. This script read the files given in the file `plot_files.lst` and will plot at the beginning all data. Further limitations of the plotted data can be done using the UI. For a plot in wavelength axis the first text box need to contain a **'w'**. For plotting only data from a subsection of files, the first text boxes 'Exclude orders/frames' and 'Exclude' need to be used. For plotting only a subsection of data types the second text boxes with 'Exclude orders/frames' and 'Exclude' are used to define this. For plotting only some orders, the last text boxes are used (see an example in Fig. 5).

5.2 Finding the radial velocity

The radial velocity measurement is based on the *CERES* pipeline. Thereby a template is cross correlated with the scientific spectrum. For the analysis the following steps need to be run:

```
ls extracted/*.fits > analysis_files.lst
python <path to script>/find_rv.py | tee RV_logfile
```

6 Extracting HARPS data

6.1 Getting the data

Search and download the data from http://archive.eso.org/eso/eso_archive_main.html. For example data select night '2015 07 30' and check only 'HARPS/LaSilla'.

The reduced data can be found at: http://archive.eso.org/wdb/wdb/eso/repro/form. For example data select night '30 07 2015'.

The raw data (*fits.Z) can be extracted with

```
gunzip *
cat <<EOT > extract_red_chip.py
#!/usr/bin/python
# -*- coding: utf-8 -*-
import os
from astropy.io import fits

os.system('mkdir -p red_chip')
for entry in os.popen('ls -1 *.fits').readlines():
    im = fits.getdata(entry[:-1],2)
    im_head = fits.getheader(entry[:-1],2)
    fits.writeto('red_chip/'+entry[:-1], im, im_head, clobber=True)
EOT
python extract_red_chip.py
```

The reduced data (*.tar) can be extracted using

```
cat *.tar | tar -xvf - -i
```

6.2 Preparing for data extraction

The following replacements are necessary in the conf.txt file:

- `subframe = [4096,2048,0,50]`
- `rotate_frame = 180`
- `bin_search_orders = 20,2`
- `arcshift_range = [-12, -20]` (red chip),
  `arcshift_range = [-12, -20]` (blue chip
- `bias_calibs_create = [subframe]`
- `bias_rawfiles = HARPS.2015-07-30T19:57:17.883.fits`
  (Header: `OBJECT = 'BIAS,BIAS'`)
- `sflat_calibs_create = subframe, bias`
- `sflat_rawfiles = HARPS.2015-07-30T19:58:10.417.fits`
  (Header: `OBJECT = 'LAMP,DARK,TUN'`)
- `arc_calibs_create = subframe, bias`
- `arc_rawfiles = HARPS.2015-07-30T23:33:11.406.fits`
  (Header: `OBJECT = 'WAVE,WAVE,THAR2'`)

- `arc_l_calibs_create = subframe, bias`
- `arc_l_rawfiles = HARPS.2015-07-30T23:33:11.406.fits, HARPS.2015-07-30T23:51:34.374.fi`
  (Header: `OBJECT = 'WAVE,WAVE,THAR2'`)
- `arc_s_calibs_create = subframe, bias`
- `arc_s_rawfiles = HARPS.2015-07-30T23:51:34.374.fits`
  (Header: `OBJECT = 'WAVE,WAVE,THAR2'`)
- `arcflat_calibs_create = subframe, bias`
- `arcflat_rawfiles = HARPS.2015-07-30T20:00:05.103.fits, HARPS.2015-07-30T20:00:44.455.`
  (Header: `OBJECT = 'LAMP,LAMP,TUN'`)
- `original_master_arc_solution_filename = master_arc_solution_first.fits`.
  The file `arc_lines_wavelength.txt` needs to be created (Section 3.4).
- `arc_lines_catalog = /home/ronny/Scripts/exohspec/reference_lines_ThAr-HARPS_Ceres.txt`
- `use_catalog_lines = NeI, UI, ThI, ArI, ?, NoID, ArII, ThII`

Afterwards, all scripts can run as described above.

A Module dependencies if not using Anaconda

This program relies on the following modules in python 2.7

- numpy
- os
- sys
- time
- datetime
- operator
- copy
- random
- warnings
- tqdm
- pickle
- json
- astropy
- scipy
- matplotlib

if python2:

- Tkinter
- collections

if python3:

- tkinter