HESP PIPELINE v. 1.0

Installation and Usage

Arun Surya
20/06/2017

1. INSTALLATION

The HESP pipeline installation zip file can be downloaded from ,
https://www.iiap.res.in/hesp/hesp_pipeline.zip .
The zip file will contain

a) **bin** directory with the source files and support files
b) **req.txt** which has the list of python packages required for the installation
c) hesp.config the config file for the pipeline

The requirement for running the hesp pipeline is Python 2.7 or above. Also for the installation of the packages, required to run the pipeline, make sure you have python-pip installed in your system. You can install it in ubuntu by

sudo apt-get install python-pip

or in Fedora by,

yum install python-pip

Use pip to install the packages listed in req.txt

sudo pip install -r req.txt

Once the packages are installed copy the bin directory to a location in your home directory. This could be inside a subdirectory also
Copy **hesp.config** to your home directory. hesp.config has to be directly in your home directory and not inside any subdirectories .

Edit your **hesp.config** and give the exact path to the bin directory in the "path=" line in Config subsection of hesp.config.

Now to make the python scripts inside the bin directory executable. Go to the bin directory and use the command

sudo chmod a+x hesp_*

In Fedora one can accomplish the same using chmod command in root #.
Now the bin directory has to be added to the linux environment variable $PATH to be accessible at different locations .
For this edit your .bash_profile or .bashrc file to add

PATH=$PATH:~/hesp/bin
export PATH

where **~/hesp/bin** is the path of the hesp source files.
Depending on .bashrc or .bash_profile you used you will need to open a new terminal or login again for the PATH variable to be active.

2. PIPELINE ROUTINES

The list of pipeline routines available for reduction are given below,

Command	Use
hesp_headerfix	Check for inconsistencies in the header and fix it for file classification and resolution mode.
hesp_headerupdate	Update headers of the files according to the new header format.
hesp_createlist	Create the ' files.txt' file with the list of files and related info used for reduction
hesp_preproc	Preprocessing of the files including, Bias Subtraction, Overscan Correction, Trimming and Cosmic Ray Correction
hesp_extract	Extract the orders from the processed files listed in 'files.txt'.
hesp_extract_file	Extract only a single file given as the argument for the command
hesp_traceview	View the order trace overplotted on the data file given as the argument.
hesp_traceshift	Adjust the trace by shifting interactively and updating the global traces in the bin directory.
hesp_addwave	Create wavelength calibrated spectra for a single file or a list of files given as argument.
hesp_recalib	Find out the global shifts in ThAr spectra and adjust and re-calibrate the wavelength solution accordingly
hesp_view	Interactive viewing of the wavelength calibrated spectra
hesp_median	Create a median combined image from the list of images given as input.

3. **USAGE WITH EXAMPLES**

To be using the hesp pipeline. Gather the files required for reduction into a directory. For the tutorial we will assume this as **~/data**

Especially for old data , it is recommended to update the headers and fix any inconsistencies in the header using **hesp_headerupdate** and **hesp_headerfix** respectively.

You will just need to run these command from the directory of your data

```
~/data:$ hesp_headerupdate
~/data:$ hesp_headerfix
```

Now one can proceed to start the reduction process. Run the **hesp_createlist** command to create the files.txt which will have the list and info of the files in the directory.

```
~/data:$ hesp_createlist
```

This will create a **files.txt** in the same directory as the data.

```
BIAS
File                    Mean      RMS       Date          Time
19Oct2016_002.fits      320.37    4.24      2016-10-19    06:43:46.248
19Oct2016_069.fits      327.14    4.09      2016-10-19    18:34:26.038
19Oct2016_001.fits      320.13    3.48      2016-10-19    06:41:30.895

OBJECT
File                    Object        Mean      Exposure  Resolution  Date          Time
19Oct2016_045.fits      HD221354      330.19    2400      High        2016-10-19    13:36:05.404
19Oct2016_053.fits      CI Cam        334.19    1200      Low         2016-10-19    16:06:51.334
19Oct2016_043.fits      HD221354      326.41    60        High        2016-10-19    12:48:55.595
19Oct2016_044.fits      HD221354      330.77    2400      High        2016-10-19    12:52:27.972
19Oct2016_055.fits      2M0826+612    326.24    60        Low         2016-10-19    17:14:40.221
19Oct2016_051.fits      CI Cam        333.36    1200      Low         2016-10-19    15:24:08.748
19Oct2016_052.fits      CI Cam        334.37    1200      Low         2016-10-19    15:45:17.863
19Oct2016_049.fits      CI Cam        594.71    60        Low         2016-10-19    15:16:34.609
19Oct2016_050.fits      CI Cam        326.15    60        Low         2016-10-19    15:19:11.320
19Oct2016_056.fits      2M0826+612    339.93    2700      Low         2016-10-19    17:17:59.842
19Oct2016_046.fits      HD221354      329.67    2400      High        2016-10-19    14:18:42.141
19Oct2016_054.fits      CI Cam        325.99    1200      Low         2016-10-19    17:06:48.863

FLAT
File                    Mean      Exposure  Resolution  Date          Time
19Oct2016_065.fits      1701.23   10        High        2016-10-19    18:26:34.010
19Oct2016_061.fits      2043.64   10        Low         2016-10-19    18:20:12.056
19Oct2016_060.fits      2046.99   10        Low         2016-10-19    18:18:31.542
19Oct2016_059.fits      2060.14   10        Low         2016-10-19    18:16:18.555
19Oct2016_063.fits      2044.62   10        Low         2016-10-19    18:23:23.806
19Oct2016_064.fits      1701.38   10        High        2016-10-19    18:25:13.985
19Oct2016_058.fits      501.82    1         Low         2016-10-19    18:13:41.326
19Oct2016_066.fits      1700.97   10        High        2016-10-19    18:27:52.487
19Oct2016_068.fits      1699.15   10        High        2016-10-19    18:30:47.614
```

Fig 1 . **files.txt** contents

The **files.txt** contains the information on the files in the directory and classification to Bias,Flat,Calib or Object frames. The **hesp_preproc** and **hesp_extract** commands takes the information from this file for the further reduction processes. So one can edit **files.txt** to control the reduction flow.

hesp_preproc command is used to do the preprocessing. It will do Overscan Correction, Bias Correction and Cosmic Ray Correction sequentially. This will also trim your raw files to the science frame. You can disable or enable each of the steps in the hesp.config file as follows.

The **hesp.config** file has **[Process]** subsection where by default Bias, Overscan and Cosmic Ray Corrections are enabled. You can disable any of it by setting either bias,overscan or cosmic variable to False.

```
[Config]
path=/home/arun/bin/

[Process]
bias=True
overscan=True
cosmic=True
csmc_clip=25

[Headers]
objname=OBJECT
dateobs=DATE-OBS
timeobs=UT
exptime=EXPTIME
objtype=OBJ-TYPE
reseltype=RES

[Keys]
reselkeylow=Low
reselkeyhigh=High
biaskey=0
flatkey=2
calibkey=3
objectkey=4

[Fits]
index=1

[Extraction]
binsize=5
binres=.5
```

Fig 2. **hesp.config** settings

You can also set the variable **csmc_clip** to adjust the sensitivity of the Cosmic Ray removal. This is set currently for optimal extraction.

Finally preprocessing can be executed by running **hesp_preproc** in the data directory.
~/data:$ **hesp_preproc**

This will create for each object, flat or calib file , a corresponding **filename_pp.fit** file**,** which is the post-processed output file.
Two routines, **hesp_traceview** and **hesp_traceshift** are provided to check the trace curves of the orders used by the pipeline. For optimal extraction these lines should be following the middle section of each order.

hesp_traceview can be used with a file argument and a keyword for the trace used ('low'/'high').

~/data:$ **hesp_traceview filename_pp.fit low**
~/data:$ **hesp_traceview filename_pp.fit high**

This will show the processed file overplotted with the trace lines.

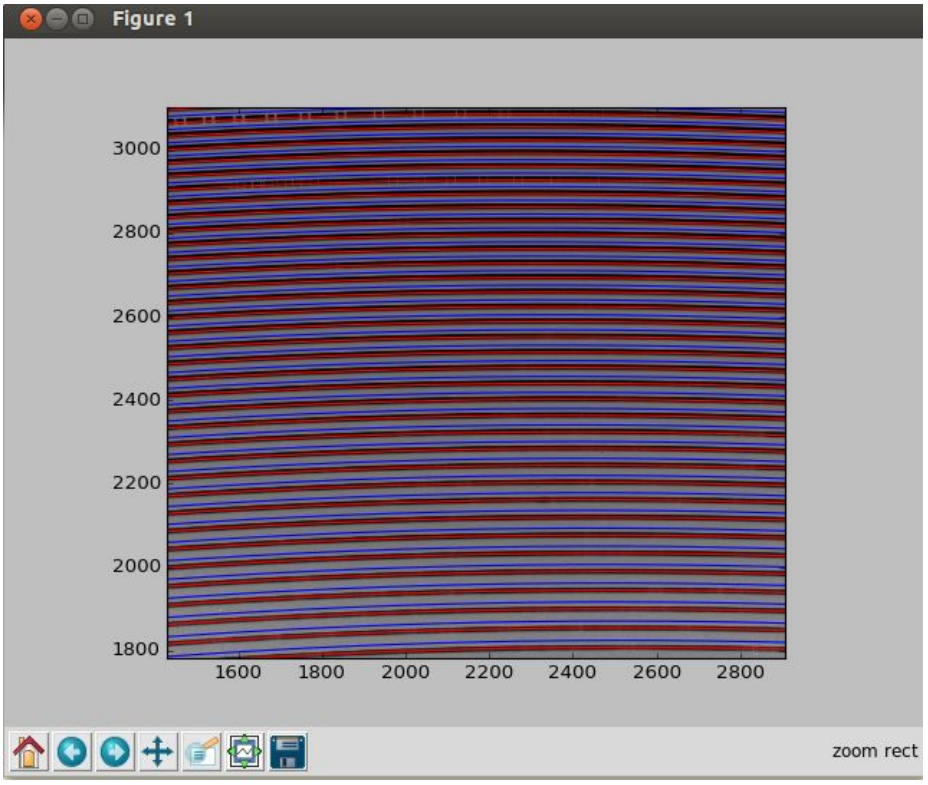

Fig 3. **hesp_traceview** interface

hesp_traceshift can be called similar to **hesp_traceview** with a file argument and a keyword for the trace used ('low'/'high').

~/data:$ **hesp_traceview filename_pp.fit low**
~/data:$ **hesp_traceview filename_pp.fit high**

But it provides with extra options to shift the traces vertically to align with the order positions on the data file. There is a slider using which the traces can be shifted. After alignment is confirmed one can save it using the 'Save' button on the interface. This will update the trace files in the hesp bin directory.

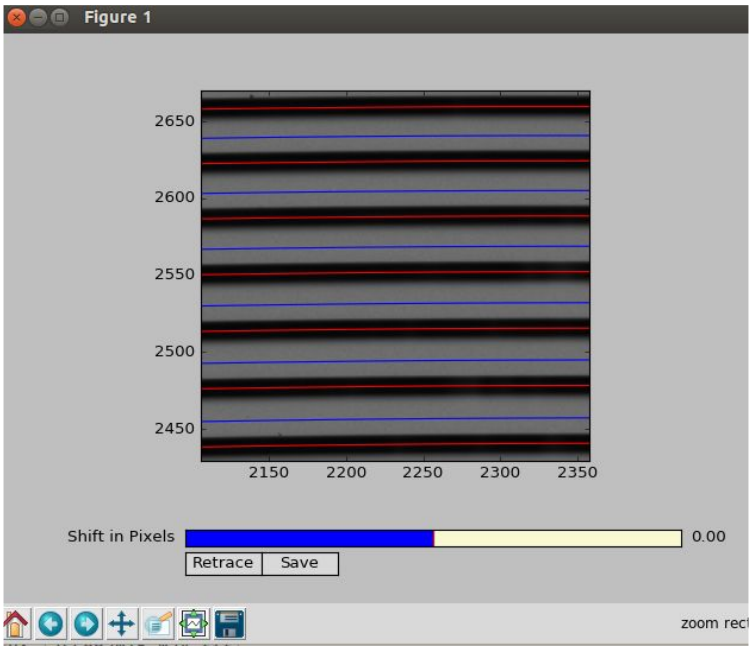

Fig 4. **hesp_traceshift** interface

To extract the orders from the preprocessed files two routines are available.
hesp_extract and **hesp_extract_file**

hesp_extract extracts the orders from the preprocessed files of all object,calib and flat files listed in **files.txt**. The extraction parameters **binsize** and **binres** can be set in hesp.config. These sets the vertical binning length and interpolation resolution used for extraction.
hesp_extract_file extracts orders for a single preprocessed file given as an argument.

The usage is as shown below,
~/data:$ **hesp_extract**
~/data:$ **hesp_extract_file filename_pp.fit** , where **filename_pp.fit** is the file to be extracted.

The extracted files are saved as **filename_fiber1_ec.fit** and **filename_fiber2_ec.fit** for each fiber of each data file.
The '**_ec.fit**' in the end of the filenames indicates the extracted spectra. These files are also in fits format compatible with IRAF echelle reduction. Thus they can be used in IRAF for wavelength calibration and other echelle routines.

hesp_addwave is the routine for wavelength calibration of the extracted spectra. The program can be in two formats. One with extracted **filename_fiber1/2_ec.fit** as the argument.

~/data:$ **hesp_addwave filename_fiber1/2_ec.fit**

It can be also used with a text file containing the names of extracted files to be calibrated using the **--list** keyword. For example if one creates **list.txt** with filenames.

filename1_fiber1/2_ec.fit
filename2_fiber1/2_ec.fit
filename3_fiber1/2_ec.fit
...

Then **hesp_addwave** can be called with
~/data:$ **hesp_addwave --list list.txt**

The routine will create **filename1_fiber1/2_wc.fit** for each extracted spectra calibrated. The **'_wc.fit'** files created have wavelength information encoded into their header in IRAF compatible format. Hence one can use IRAF tools like **splot** to further process the file.

hesp_recalib allows to calibrate the wavelength solution using a ThAr calibration file in the directory. The format for calling the routine is as below,

~/data:$ **hesp_recalib filename1_fiber1/2_ec.fit high/low fib1/fib2**

The routine will cross correlate a sample of orders to find a global shift in the ThAr spectra and use it to adjust the wavelength solution accordingly.

The default wavelength solution can be restored using,

~/data:$ **hesp_recalib --set-default high/low fib1/fib2**

Routine **hesp_view** will allow to view the full wavelength calibrated spectra with an interactive Matplotlib GUI.

~/data:$ **hesp_view filename1_fiber1/2_ec.fit** , or
~/data:$ **hesp_view filename1_fiber1/2_wc.fit**

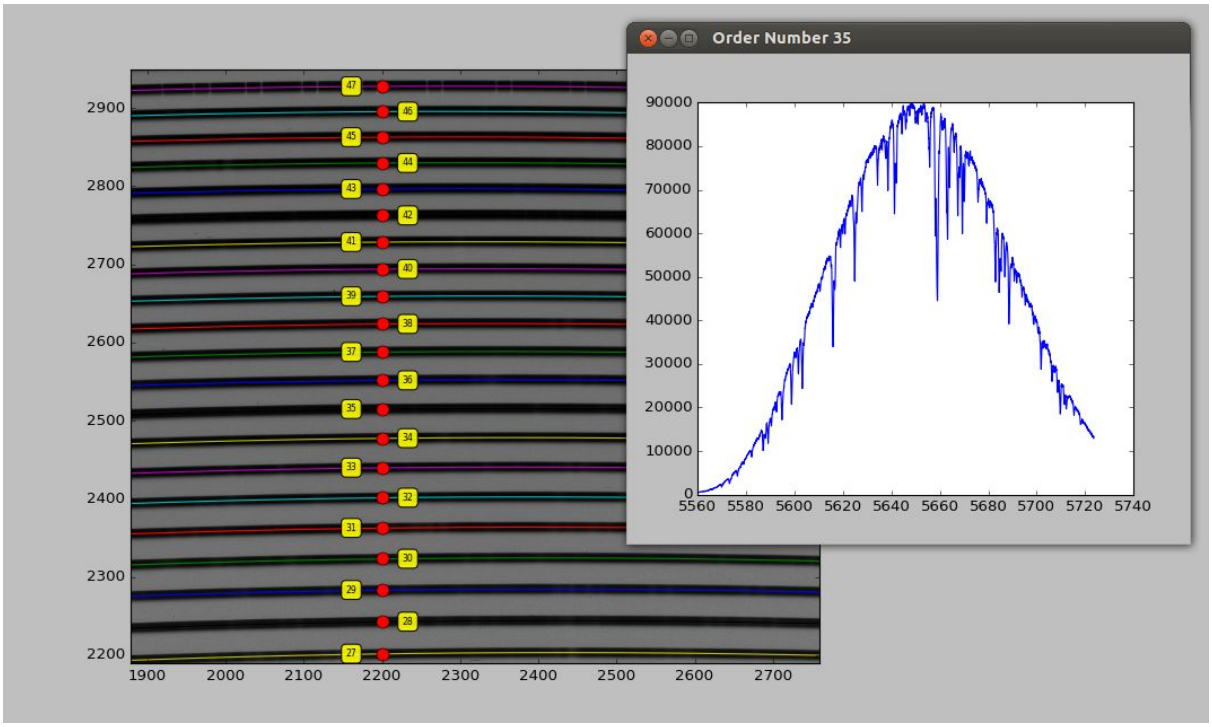

Fig 5. **hesp_view** interface

Utility routine **hesp_median** is available to median combine a list of files. The format is as below,

~/data:$ hesp_median list.txt ,

where **list.txt** contains names of the files to be median combined.