pyraf-dbsp Documentation

Release 0.2.1

Eric Bellm and Branimir Sesar

Sep 06, 2018

Overview

A Spectral Reduction Pipeline for the Palomar Double Beam Spectrograph

CHAPTER 1

Quick Start/Command Summary

Complete night reduction:

`cd` to your data directory.

Save a copy of your files in a "raw" directory–this code overwrites the originals!

```
mdkir raw
cp *.fits raw
```

Start `ipython` and load the script:

```
%run /path/to/dbsp.py
```

For users on the Caltech astro network, log in to soroban and execute:

```
export PATH="/scr/ebellm/anaconda/bin:$PATH"
source activate iraf27
mkiraf  # choose xgterm
ipython
%run /home/ebellm/observing/reduction/dbsp/dbsp.py
```

Exclude any images that you don't want to analyze (use your log; especially focus/test exposures from the beginning of the night):

```
mark_bad([47,49,50],side='blue')
mark_bad([35],side='red')
```

Create arcs and dome flats (run this after you have at least one science exposure):

```
create_arc_dome()
```

Process flux calibration standards, if desired. If not, skip this step and set `flux=False` in `extract1D()`.

```
store_standards([41,42,43], side='blue')
```

Extract data:

```
extract1D(61,side='blue')
```

For basic telluric correction on the red side, first extract an appropriate telluric calibrator, then pass it to `store_standards` and `extract1D`:

```
extract1D(77,side='red', Flux=False)

store_standards([41,42,43], side='red', telluric_cal_id = 77)

extract1D(63,side='red',flux=True, telluric_cal_id = 77)
```

To process a large number of science spectra in a row:

```
batch_process(20, 45, side='blue', quicklook='no')
```

Finally, join spectra from the blue and red sides, identify pairs (or more) of images:

```
combine_sides([61],[63,64])
```

CHAPTER 2

Using the pipeline

This pipeline provides a thin wrapper around pyraf, which in turn is a thin wrapper around IRAF. Accordingly, IRAF documentation for functions like `doslit`, `autoidentify`, `standard`, and `sensfunc` may be helpful to understand the command options. Search here for IRAF function documentation.

You can obtain thorough help on any of the DBSP pipeline commands defined in this package with `help()`, e.g., `help(extract1D)`.

For help on IRAF commands, explore some of the following options: Click graphics window, hit `?` For window commands: click graphics window, type `w ?`. For function documentation, type at the `ipython` prompt:

```
iraf.help('doslit')
iraf.dir('onedstds$')
iraf.type('onedstds$README') (or iraf.page)
iraf.epar('doslit')
```

`mark_bad()`:

All image files are uniquely identified by their `imgID` (an integer number) and the spectrograph arm (`side='red'` or `'blue'`), e.g., `red0018.fits`. Most functions expect either a single `imgID` or a python list.

`create_arc_dome()`:

This command should operate automatically and generate several calibration products in the current directory.

After running, look at the flats (e.g, `flat_blue_1.0.fits`, `raw_flat_blue_1.0.fits`) in ds9 before continuing to make sure they don't have any weird features. While rare, these issues are typically the result of science exposures being incorrectly detected as dome flats due to header keyword problems.

`store_standards()`:

This function extracts 1D spectra for the standard star exposures using `extract1D()`, so review the key commands below.

`extract1D()`:

This function is a convenience wrapper for IRAF's `doslit` and several other functions. The pipeline loads a modified version of `doslit` to minimize unneeded interactivity.

If you set `quicklook='yes'`, it will proceed without any intervention. The default, `quicklook='no'`, prompts you to identify and fit the trace and fit the dispersion function. This requires you to know which key commands the IRAF graphics window is expecting. This tutorial provides some guidance, and the discussion below identifies the most important.

When IRAF is waiting for your input in the graphics window, you can type `?` to get a list of the commands. Some of these will begin with a colon, meaning that you can change a value by clicking on the gray bar in the bottom of the graphics window and typing something like `:function spline3`. Typically once you are satified with a given selection you type `q` to accept and continue.

Note that commands in the graphics window may require you to click to focus; then you may have to click back to type responses in the terminal.

The first screen allows you to edit the aperture for extracting the spectrum:

- Use `d` to delete trace, `m` to set it
- `b` to enter background editing
- `z` to delete background intervals
- `s s` (with cursor positions) to mark new fit regions
- `f` to fit
- `q` to quit

The second screen allows you to fit the trace interactively using IRAF's `icfit`:

- `?` for help
- `:order #` to change fit order. Default of 4 is probably fine; don't worry too much about excursions on the faint ends of the trace
- `d` to delete any points biasing the fit
- `s` (twice) to change the sampling region (e.g., to exclude faint ends)
- `f` to refit

Aim for RMS < 0.07 or so.

The next screen lets you fit the wavelength solution with IRAF's `autoidentify` and `reidentify`:

- `f` to fit/refit
- `j` to switch to residuals plot
- `d` to delete outliers
- `q` twice to save solution

Blue side can be bad below 4000: delete all the points with `d`, hit `q`, `f`, `l` (to reidentify lines). Good RMS is < 0.07 or so. `q` to save and continue.

If you have to (or want to) manually identify lines:

- `w x` to zoom in
- `m` to specify a line (with cursor hovering over it): enter the wavelength in angstroms
- `w a` to zoom out

do it for a second line (may already be identified)

- `f` to fit
- `d` to delete outliers

- `l` or `y` to ask for more lines
- `f` to fit again
- `q` twice to save solution

When reidentifying, bad RMS may often be fixed just by hitting `l` then `q`.

The process above (choose aperture, fit trace, fit/verify dispersion solution) occurs each time you run `extract1D()`. If the extraction is part of the `store_standards()` call, you'll then continue with the following prompts:

The terminal will prompt: `change wavelength coordinate assignments?` This is your chance to set wavelength range & binning: type `yes`. Sensible defaults (assuming the 600/4000 grating on the blue side and 316/7500 grating on the red):

```
5500-10000, 1.525 red (new red camera)
5500-7800, 2.47 red (old red camera)
3800-5700, 1.07 blue
```

Accept the suggested number of output pixels, then say `no` when it asks you again if you want to change them.

Next, you'll fit throughput functions for the standards; this uses IRAF's `standards`.

- `?` for list or `iraf.page('onedstds$README')`

all of the commonly-used ones are in `onedstds#iidscal`:

```
g191b2b
feige34
bd332642
bd284211
```

edit bandpasses (choose smooth regions):

- `a a` (with mouse pointer at two positions) to place new bands
- `d` to delete bands (on absorption features, say)
- `q` to quit and save

You'll define bands for all of your standard exposures, then fit the sensitivity function with IRAF's `sensfunc`.

- `?` for help
- `s` over graphs to eliminate mean shifts due to non-photometric conditions (toggles)
- `d` to delete bad points

Make sure the fitted function doesn't go up after the last points–it will blow up the noise. Also consider decreasing the order of the fit (`:order 4`) to avoid spline artifacts

After completing the above steps, the sensitivity function will be defined and you'll be ready to obtain science spectra using `extract1D()`.

`combine_sides()`:

This function combines data from the two spectrograph arms; it can also coadd data in an uncertainty-weighted manner. The output filenames are taken from the `OBJECT` header keyword.

CHAPTER 3

Tips for on the fly reduction

Start reducing data after you have taken your first standard star exposure.

Copy (or `rsync`) new data into your `raw` subdirectory; then call `sync()` to bring the new files into your working directory without overwriting those you've already processed.

You can set `quicklook=yes` in `extract1D` to skip manual identification and fitting of the trace and dispersion. Use with caution, particularly with faint sources.

CHAPTER 4

FAQ

Why isn't autoidentify putting the arc lines in the right place? Why is my wavelength solution bad?

Are the crval and cdelt values appropriate for the CCD, grating, and angle you're using? The `check_gratings_angles()` function attempts to automatically determine the correct values, but it is not foolproof.

You can use this calculator to determine approximate `crval` (center wavelength in angstroms) and `cdelt` values (disperson in Angstroms/pixel) for your grating. Input the grating and side you are using and a guess for the center wavelength; click calculate and note the reported grating angle. Edit your center wavelength guess until you find your grating angle. The calculator then gives the center wavelength and dispersion you want.

reset to new values with:

```
det_pars['red']['crval'] = 7330
det_pars['red']['cdelt'] = 3.022
```

I don't have any 0.5" slit arcs–what can I do?

Choose another slit width and set it as the default:

```
# use 1.0 arcsecond slit for arcs
create_arc_dome(arcslit='1.0')

# set default arc files
det_pars['blue']['arc']  = 'FeAr_1.0.fits'
det_pars['red']['arc']  = 'HeNeAr_1.0.fits
```

I'm getting "ERROR (1, "image keyword AIRMASS not found")" when I run `create_arc_dome()`.

One of your images is missing a header keyword–find it and mark it bad.

I'm not happy with the fluxing–what can I do?

Run `iraf.sensfunc()` and tweak to test, then repeat your `extract1D()` call.

During fluxing, iraf.calibrate crashes and says it can't find sens-blue.0001.fits. (or sens-red)

You can work around this by symlinking sens-blue.fits to sens-blue.0001.fits:

ln -s sens-blue.fits sens-blue.0001.fits

Is there any way to reduce the repetive prompting?

`pyraf-dbsp` comes with a modified doslit version that minimizes prompting.However, it seems to cause iraf errors for some users. To run it, set

```
export USEDBSPDOSLIT=1
```

before starting ipython.

I'm getting an error:

```
ERROR: An unexpected error occurred while tokenizing input
The following traceback may be corrupted or invalid
The error message is: ('EOF in multi-line statement', (11, 0))

File "<tokenize>", line 2
    if (Vars.dispcor and Vars.fluxcal1):
IndentationError: unindent does not match any outer indentation level
```

This error is mysterious; it pops up irregularly, particularly if a routine has aborted unnaturally. Try restarting `ipython` and running your command again, but be aware that in some cases it may be necessary to wipe the directory and restart from the raw images.

CHAPTER 5

Known Problems & Limitations

- expects FeAr and (simulataneous) HeNeAr arcs in the 0.5 arcsec slit
- any absorption features in the “featureless” telluric calibration will become emission features in the corrected spectrum
- fluxing is not correcting for extinction?
- only one object can be extracted from a given image
- some fluxing weirdness at the shortest blue wavelengths
- joining red and blue sides with combine_sides often introduces artifacts

CHAPTER 6

Todos

- fluxing should correct for extinction
- improve telluric correction defaults
- improve robustness of joining red and blue sides (bad fluxing kills it)
- wrapper for single extract-combine-plot run
- script to “undo” various parts of the analysis? eg, start from scratch w/ standards
- Brani suggests only using arcs taken in a single batch–need to adjust code logic
- automate wavelength coordinate assignment for autoidentify?
- calculate gain & readnoise from cal files
- set fwhm_arc correctly (currently only uses a default)
- write auto-joiner to speed joining and coaddition
- write a log file to record processing steps taken
- add pipeline version and reducer information to output headers

CHAPTER 7

Revisions

7.1 0.2.1 dev

(ongoing)

Bug fixes:

- allow check_gratings_angles to handle decimal ANGLE keywords
- handle bug where flatcombine expected multiextension FITS
- Provide smoother handling if create_arc_dome(side='both') is called when data from only one side are present

Enhancements:

- harmonize aperture formats across routines

7.2 0.2.0

April 18, 2014

Enhancements:

- detect and attempt to automatically estimate dispersion parameters for any grating and angle
- add quicklook reduction mode
- add batch_process interface
- reduce repetitive prompting in `doslit` and `calibrate`
- add license
- expand documentation

Bug fixes:

- avoid instrument translation file problem in IRAF 2.16.1

7.3 0.1.0

July 3, 2013

Initial public release.

CHAPTER 8

Function documentation

- genindex
- modindex
- search