VDRP Documentation

Release 1.0.5.post0

Jan Snigula

Feb 20, 2019

Contents

Version: 1.0.5.post0

VDRP the Virus Data Reduction Pipeline is a collection of scripts and FORTRAN programs for astrometry, throughput and flux limit calculation.

VDRP currently supports only python 2.7.

CHAPTER 1

User documentation

1.1 Astrometry routines

1.2 Throughput routines

1.3 Fluxlimit routines

1.3.1 Setting up and running the fluxlimit calculations

To calculate the fluxlimit cube of a given night shot call:

```
vdrp_setup_flim night shot
```

This will create a subdirectory tree of the form `nightvshot/flim` and in there a slurm batch script named `flimnightvshot.slurm` and the corresponding input files. Running the script as

```
vdrp_setup_flim --commit night shot
```

the slurm script will be sent to the batch system automatically. If needed the default runtime of `06:00:00` can be modified using –runtime on the command line.

1.4 Spectral line extraction routines

CHAPTER 2

Developer documentation

2.1 Contribute to VDRP

2.1.1 How To

The suggested workflow for implementing bug fixes and/or new features is the following:

- Identify or, if necessary, add to our redmine issue tracker one or more issues to tackle. Multiple issues can be addressed together if they belong together. Assign the issues to yourself.
- Create a new branch from the trunk with a name either referring to the topic or the issue to solve. E.g. if you need to add a new executable, tracked by issue #1111 `do_something`:

```
svn cp ^/trunk ^/branches/do_something_1111\
-m 'create branch to solve issue #1111'
```

- Switch to the branch:

```
svn switch ^/branches/do_something_1111
```

- Implement the required changes and don't forget to track your progress on redmine. If the feature/bug fix requires a large amount of time, we suggest, when possible, to avoid one big commit at the end in favour of smaller commits. In this way, in case of breakages, is easier to traverse the branch history and find the offending code. For each commit you should add an entry in the `Changelog` file.

  If you work on multiple issues on the same branch, close one issue before proceeding to the next. When closing one issue is good habit to add in the description on the redmine the revision that resolves it.

- Every function or class added or modified should be adequately documented as described in *Coding style*.

  Documentation is essential both for users and for your fellow developers to understand the scope and signature of functions and classes. If a new module is added, it should be also added to the documentation in the appropriate place. See the existing documentation for examples.

  Each executable should be documented and its description should contain enough information and examples to allow users to easily run it.

- Every functionality should be thoroughly tested for python 3.5 or 3.6 in order to ensure that the code behaves as expected and that future modifications will not break existing functionalities. When fixing bugs, add tests to ensure that the bug will not repeat. For more information see *Testing*.
- Once the issue(s) are solved and the branch is ready, merge any pending change **from** the trunk:

```
svn merge ^/trunk
```

  While doing the merge, you might be asked to manually resolve one or more conflicts. Once all the conflicts have been solved, commit the changes with a meaningful commit message, e.g.: `merge ^/trunk into ^/branches/do_something_1111`. Then rerun the test suite to make sure your changes do not break functionalities implemented while you were working on your branch.

- Then contact the maintainer of `fplaneserver` and ask to merge your branch **back to the trunk**.

Information about branching and merging can be found in the svn book. For any questions or if you need support do not hesitate to contact the maintainer or the other developers.

2.1.2 Coding style

All the code should be compliant with the official python style guidelines described in **PEP 8**. To help you keep the code in spec, we suggest to install plugins that check the code for you, like Synstastic for vim or flycheck for Emacs.

The code should also be thoroughly documented using the numpy style. See the existing documentation for examples.

2.1.3 Testing

Note: Every part of the code should be tested and should run at least under python 3.5 and possibly 3.6

`fplaneserver` uses the testing framework provided by the robot framework package. The tests should cover every aspect of a function or method. If exceptions are explicitly raised, this should also tested to ensure that the implementation behaves as expected.

The preferred way to run the tests is using tox, an automatised test help package. If you have installed tox, with e.g. `pip install tox`, you can run it by typing:

```
tox
```

It will take care of creating virtual environments for every supported version of python, if it exists on the system, install `fplaneserver`, its dependences and the packages necessary to run the tests and runs `py.test`

You can run the tests for a specific python version using:

```
python -m robot
```

A code coverage report is also created and can be visualized opening into a browser `cover/index.html`.

Besides running the tests, the `tox` command also builds, by default, the documentation and collates the coverage tests from the various python interpreters and can copy then to some directory. To do the latter create, if necessary, the configuration file `~/.config/little_deploy.cfg` and add to it a section called `fplaneserver` with either one or both of the following options:

```
[fplaneserver]
# if given the deploys the documentation to the given dir
doc = /path/to/dir
```

(continues on next page)

(continued from previous page)

```
# if given the deploys the coverage report to the given dir
cover = /path/to/other/dir

# it's also possible to insert the project name and the type of the document
# to deploy using the {project} and {type_} placeholders. E.g
# cover = /path/to/dir/{project}_{type_}
# will be expanded to /path/to/dir/fplaneserver_cover
```

For more information about the configuration file check little_deploy.

2.1.4 Documentation

To build the documentation you need the additional dependences described in pydep. They can be installed by hand or during `fplaneserver` installation by executing one of the following commands on a local copy:

```
pip install /path/to/fplaneserver[doc]
pip install /path/to/fplaneserver[livedoc]
```

The first install `sphinx`, the `alabaster` theme and the `numpydoc` extension; the second also installs `sphinx-autobuild`.

To build the documentation in html format go to the `doc` directory and run:

```
make html
```

The output is saved in `_doc/build/html`. For the full list of available targets type `make help`.

If you are updating the documentation and want avoid the `edit-compile-browser refresh` cycle, and you have installed `sphinx-autobuild`, type:

```
make livehtml
```

then visit http://127.0.0.1:8000. The html documentation is automatically rebuilt after every change of the source and the browser reloaded.

Please make sure that every module in `fplaneserver` is present in the *Code documentation*.

2.2 Code documentation

2.2.1 `astrometry` - Astrometry routines

Astrometry routine

Module to add astrometry to HETDEX catalgoues and images Contains python translation of Karl Gebhardt

`vdrp.astrometry.`**`add_ifu_xy`**(*wdir*, *offset_exposure_indices*)

Adds IFU x y information to stars used for matching, and save to xy_expNN.dat. Requires: getoff.out, radec2.dat Analogous to rastrom3.

Parameters

wdir [str] Work directory.

offset_exposure_indices [list] List of exposure indices to consider.

vdrp.astrometry.**add_ra_dec**(*wdir*, *als_data*, *ra*, *dec*, *pa*, *fp*, *radec_outfile='tmp.csv'*)
Call add_ra_dec to compute for detections in IFU space the corresponding RA/DEC coordinates.

New version, direct python call to pyheted.coordinates.tangent_projection.

Requires, fplane.txt Creates primarely EXPOSURE_tmp.csv but also radec.dat.

Parameters

wdir [str] Work directory.

als_data [dict] Dictionary with als data for each IFU slot.

ra [float] Focal plane center RA.

dec [float] Focal plane center Dec.

pa [float] Positions angle.

fp [FPlane] Focal plane object.

radec_outfile [str] Filename that will contain output from add_ra_dec (gets overwritten!).

vdrp.astrometry.**combine_radec**(*wdir*, *dither_offsets*, *PLOT=True*)
Computes - based on the RA Dec information of the individual exposures (from radec2_exp0?.dat) the final RA/Dec for the shot.

Parameters

wdir [str] Work directory.

Notes

Creates radec2_final.dat. Optionally create a plot indicating the individual exposure positions.

vdrp.astrometry.**compute_offset**(*wdir*, *prefixes*, *getoff2_radii*, *add_radec_angoff_trial*, *add_radec_angoff*, *add_radec_angoff_trial_dir*, *offset_exposure_indices*, *final_ang_offset=None*, *shout_ifustars='shout.ifustars'*, *ra0=None*, *dec0=None*)
Requires, fplane.txt and radec.orig. If not ra, dec are passed explicitly then the values from radec.orig are used. The pa value from radec.orig is used in any case. Creates primarely EXPOSURE_tmp.csv but also radec2.dat.

Compute offset in RA DEC by matching detected stars in IFUs against the shuffle profived RA DEC coordinates.

Parameters

wdir [str] Work directory.

prefixes [list] List file name prefixes for the collapsed IFU images.

getoff2_radii [list] List of matching radii for astrometric offset measurement.

add_radec_angoff_trial [list] Trial values for angular offsets.

add_radec_angoff [float] Angular offset to add during conversion of x/y coordinate to RA/Dec.

add_radec_angoff_trial_dir [str] Directory to save results of angular offset trials.

offset_exposure_indices [list] Exposure indices.

final_ang_offset [float] Final angular offset to use. This overwrites the values in add_radec_angoff and add_radec_angoff_trial

shout_ifustars [str] Shuffle output catalog of IFU stars.

ra0 [float] Optionally allows to overwrite use of RA from radec.orig

dec0 [float] Optionally allows to overwrite use of DEC from radec.orig

Notes

Analogous to rastrom3. Creates radec.dat, radec2.dat and radec_TRIAL_OFFSET_ANGLE.dat, radec_TRIAL_OFFSET_ANGLE2.dat.

vdrp.astrometry.**compute_optimal_ang_off**(*wdir*, *smoothing=0.05*, *PLOT=True*)
Computes the optimal angular offset angle by findin the minimal RMS of a set of different trial angles.

Takes (if exist) all three different exposures into account and computes weighted average ange (weighted by number of stars that went into the fit).

The RMS(ang_off) are interpolate with a smoothing spline. The smoothing value is a parameter to this function.

Parameters

wdir [str] Directory that holds the angular offset trials (e.g. 20180611v017/add_radec_angoff_trial)

Returns

float [Optimal offset angle.]

vdrp.astrometry.**cp_addin_files**(*wdir*, *addin_dir*, *subdir='coords'*)
Copies `addin` files. These are essentially the IFUcen files in a different format.

Parameters

addin_dir [str] Directory where the *.addin files are stored.

wdir [str] Work directory.

vdrp.astrometry.**cp_ixy_files**(*wdir*, *ixy_dir*, *subdir='coords'*)
Copies `ixy` files. These are essentially the IFUcen files in a different format.

Parameters

ixy_dir_dir :str Directory where the *.ixy files are stored.

wdir [str] Work directory.

vdrp.astrometry.**cp_post_stamps**(*wdir*, *reduction_dir*, *night*, *shotid*)
Copy CoFeS (collapsed IFU images).

Parameters

wdir [str] Work directory.

reduction_dir [str] Directory that holds panacea reductions.

night [str] Night (e.g. 20180611)

shotid [str] ID of shot (e.g. 017)

Raises

Exception

vdrp.astrometry.**cp_results**(*tmp_dir*, *results_dir*)
Copies all relevant result files from tmp_dir results_dir.

Parameters

tmp_dir [str] Temporary work directory.

results_dir [str] Final directory for results.

vdrp.astrometry.**daophot_find**(*wdir*, *prefixes*, *daophot_opt*, *daophot_sigma*, *daophot_xmin*, *daophot_xmax*, *daophot_ymin*, *daophot_ymix*)

Run initial daophot find.

Parameters

wdir [str] Work directory.

prefixes [list] List file name prefixes for the collapsed IFU images.

daophot_opt [str] Daphot sigma value.

daophot_sigma [float] Filename for daophot configuration.

daophot_xmin [float] X limit for daophot detections.

daophot_xmax [float] X limit for daophot detections.

daophot_ymin [float] Y limit for daophot detections.

daophot_ymix [float] Y limit for daophot detections.

vdrp.astrometry.**daophot_phot_and_allstar**(*wdir*, *prefixes*, *daophot_photo_opt*, *daophot_allstar_opt*, *daophot_phot_psf*)

Runs daophot photo and allstar on all IFU postage stamps. Produces *.ap and *.als files. Analogous to run4a.

Parameters

wdir [str] Work directory.

prefixes [list] List file name prefixes for the collapsed IFU images.

daophot_opt [str] Filename for daophot configuration.

daophot_photo_opt [str] Filename for daophot photo task configuration.

daophot_allstar_opt [str] Filename for daophot allstar task configuration.

vdrp.astrometry.**flux_norm**(*wdir*, *mag_max*, *infile='all.raw'*, *outfile='norm.dat'*)

Reads all.raw and compute relative flux normalisation for the three exposures.

Parameters

wdir [str] Work directory.

mag_max [float] Magnitude limit for flux normalisation.

infile [str] Output file of daomaster.

outfile [str] Filename for result file.

Notes

Analogous to run9.

vdrp.astrometry.**getDefaults**()

vdrp.astrometry.**getNorm**(*all_raw*, *mag_max*)

Comutes the actual normalisation for flux_norm.

Parameters

all_raw [str] Output file name of daomaster, usuall all.raw.

mag_max [float] Magnitude cutoff for normalisation. Fainter objects will be ignored.

Notes

Analogous to run9.

vdrp.astrometry.**get_active_slots**(*wdir*, *reduction_dir*, *exposures*, *night*, *shotid*)
Figures out which IFU slots actually delivered data, by checking if a corresponding multifits exists for all exposures in a shot.

vdrp.astrometry.**get_als_files**(*fp*, *exp_prefixes*)
Derives for a list of exposure prefixes a list of *.als files, but rejects any that refer to an IFU slot which is not contained in the fplane.

Parameters

fp [pyhetdex.het.fplane.FPlane] Fplane object.

exp_prefixes [list] List of epxosure prefixes.

Returns

(list): List of *.als files.

vdrp.astrometry.**get_exposures**(*prefixes*)
Computes unique list of exposures from prefixes.

Parameters

args [argparse.Namespace] Parsed configuration parameters.

prefixes [list] List file name prefixes for the collapsed IFU images

Returns

(list): Unique list of exposure strings.

vdrp.astrometry.**get_exposures_files**(*basedir*)
Create list of all file prefixes based on the existing collapsed IFU files in the current directory.

From files:

20180611T054545_015.fits ... 20180611T054545_106.fits 20180611T055249_015.fits ... 20180611T055249_106.fits 20180611T060006_015.fits ... 20180611T060006_106.fits

Creates:

{ 'exp01' : ['20180611T054545_015',...,'20180611T054545_106'] 'exp02' : ['20180611T055249_015',...,'20180611T055249_106'] 'exp03' : ['20180611T060006_015',...,'20180611T060006_106']

}

Parameters

basedir [str] Directory to search.

Returns

OrderedDict [Ordered dictionary with pairs of exposure] string "exp??" and time and list of

vdrp.astrometry.**get_fiber_coords**(*wdir*, *active_slots*, *dither_offsets*, *subdir='coords'*)
Calls add_ra_dec for all IFU slots and all dithers.

The is the main routine for getcoord which computes the on-sky positions for all fibers.

Essentially this is a whole bunch of calls like.:

add_ra_dec –ftype line_detect –astrometry 262.496605 33.194212 262.975922 –fplane /work/00115/gebhardt/maverick/sci/panacea/shifts/fplane.txt –ihmps 015 –fout i015_1.csv –dx 0 –dy 0 015.addin

... add_ra_dec –ftype line_detect –astrometry 262.496605 33.194212 262.975922

–fplane /work/00115/gebhardt/maverick/sci/panacea/shifts/fplane.txt –ihmps 015 –fout i015_2.csv –dy 1.27 –dx -0.73 015.addin

... add_ra_dec –ftype line_detect –astrometry 262.496605 33.194212 262.975922

–fplane /work/00115/gebhardt/maverick/sci/panacea/shifts/fplane.txt –ihmps 015 –fout i015_3.csv –dy 1.27 –dx 0.73 015.addin

Parameters

wdir [str] Work directory.

Notes

This creates a list of files iIFUSLOT_DITHERNUM.csv that store the on-sky fiber coordinates.

vdrp.astrometry.**get_prefixes**(*wdir*)
Create list of all file prefixes based on the existing collapsed IFU files in the current directory.

Parameters

wdir [str] Work directory.

vdrp.astrometry.**get_ra_dec_orig**(*wdir*, *reduction_dir*, *night*, *shotid*, *user_pa=-999999.0*)
Reads first of the many multi* file'd headers to get the RA, DEC, PA guess from the telescope.

Parameters

wdir [str] Work directory.

reduction_dir [str] Directory that holds panacea reductions.

night [str] Night (e.g. 20180611)

shotid [str] ID of shot (e.g. 017)

Notes

Creates radec.orig

vdrp.astrometry.**get_track**(*wdir*, *reduction_dir*, *night*, *shotid*)
Reads first of the many multi* file'd headers to get the track.

Parameters

wdir [str] Work directory.

reduction_dir [str] Directory that holds panacea reductions.

night [str] Night (e.g. 20180611)

shotid [str] ID of shot (e.g. 017)

Returns

—— (int): 0 = east track, 1 = west track

Notes

This function is so emparrisingly similar to get_ra_dec_orig that they should probably be combined.

vdrp.astrometry.**load_als_data**(*als_files*)
Load set of als files.

Parameters

als_files [list] List of file names.

Returns

—— (OrderedDict): Dictionary with als data for each IFU slot.

vdrp.astrometry.**main**(*args*)
Main function.

vdrp.astrometry.**mk_dithall**(*wdir*, *active_slots*, *reduction_dir*, *night*, *shotid*, *subdir='.'*)
This creates the dithall.use file that is required by the downstream processing functions like photometry and detect.

The file dithall.use contains for every exposure (1-3) and every fiber the RA/Dec on sky coordinats and the multifits file where the spectrum is stored and the fiber number.

vdrp.astrometry.**mk_match_matrix**(*wdir*, *ax*, *exp*, *image_files*, *fplane_file*, *shout_ifu_file*, *xy_file*, *radec_file*)
Creates the actual match plot for a specific exposures. This is a subroutine to mk_match_plots.

Parameters

wdir [str] Work directory.

prefixes [list] List file name prefixes for the collapsed IFU images.

ax [pyplot.axes] Axes object to plot into.

exp [str] Exposure string (e.g. exp01)

image_files [list] List of file names.

fplane_file [str] Focal plane file filename.

shout_ifu_file [str] Shuffle IFU star catalog output filename.

xy_file [str] Filename for list of matched stars, aka xy_exp??.dat.

radec_file [str] File that contains shot ra dec position.

vdrp.astrometry.**mk_match_plots**(*wdir*, *prefixes*)
Creates match plots.

Parameters

wdir [str] Work directory.

prefixes [list] List file name prefixes for the collapsed IFU images.

vdrp.astrometry.**mk_post_stamp_matrix**(*wdir*, *prefixes*, *cofes_vis_vmin*, *cofes_vis_vmax*)
Create the IFU postage stamp matrix image.

Parameters

wdir [str] Work directory.

prefixes [list] List file name prefixes for the collapsed IFU images.

cofes_vis_vmin [float] Minimum value (= black) for matrix overview plot.

cofes_vis_vmax [float] Maximum value (= black) for matrix overview plot.

vdrp.astrometry.**mkmosaic**(*wdir*, *prefixes*, *night*, *shotid*, *mkmosaic_angoff*)
Creates mosaic fits image.

Parameters

wdir [str] Work directory.

prefixes [list] List file name prefixes for the collapsed IFU images.

night [str] Night (e.g. 20180611)

shotid [str] ID of shot (e.g. 017)

mkmosaic_angoff [float] Angular offset to add for creation of mosaic image.

vdrp.astrometry.**mktot**(*wdir*, *prefixes*, *mktot_ifu_grid*, *mktot_magmin*, *mktot_magmax*, *mktot_xmin*, *mktot_xmax*, *mktot_ymin*, *mktot_ymax*, *dither_offsets*)
Reads all *.als files. Put detections on a grid corresponding to the IFU position in the focal plane as defined in config/ifu_grid.txt (should later become fplane.txt. Then produces all.mch.

Parameters

wdir [str] Work directory.

prefixes [list] List file name prefixes for the collapsed IFU images.

mktot_ifu_grid [str] Name of file that holds gird of IFUs offset fit (mktot).

mktot_magmin [float] Magnitude limit for offset fit.

mktot_magmax [float] Magnitude limit for offset fit.

mktot_xmin [float] X limit for offset fit.

mktot_xmax [float] X limit for offset fit.

mktot_ymin [float] Y limit for offset fit.

mktot_ymax [float] Y limit for offset fit.

Notes

Analogous to run6 and run6b.

vdrp.astrometry.**parseArgs**(*args*)
Parses configuration file and command line arguments. Command line arguments overwrite configuration file settiongs which in turn overwrite default values.

Parameters

args [argparse.Namespace] Return the populated namespace.

vdrp.astrometry.**project_xy**(*wdir*, *radec_file*, *fplane_file*, *ra*, *dec*)
Translate *all* catalog stars to x/y to display then and to see which ones got matched. Call pyhetdex tangent_plane's functionality to project ra,dec to x,y.

Parameters

wdir [str] Work directory.

radec_file [str] File that contains shot ra dec position.

fplane_file [str] Focal plane file filename.

ra [list] List of ra positions (in float, degree).

dec [list] List of dec positions (in float, degree).

vdrp.astrometry.**redo_shuffle**(*wdir*, *ra*, *dec*, *track*, *acam_magadd*, *wfs1_magadd*, *wfs2_magadd*, *shuffle_cfg*, *fplane_txt*, *night*, *catalog=None*)

Reruns shuffle to obtain catalog of IFU stars.

Creates a number of output files, most importantly *shout.ifustars* which is used as catalog for the offset computation.

Parameters

wdir [str] Work directory.

ra [float] Right ascension in degrees.

dec [float] Declination in degrees.

track [int] East or west track (0, 1)

acam_magadd [float] do_shuffle acam magadd.

wfs1_magadd [float] do_shuffle wfs1 magadd.

wfs2_magadd [float] do_shuffle wfs2 magadd.

vdrp.astrometry.**rmaster**(*wdir*)

Executes daomaster. This registers the sets of detections for the thre different exposrues with respec to each other.

Parameters

wdir [str] Work directory.

Notes

Analogous to run8b.

vdrp.astrometry.**run**()

2.2.2 `calc_fluxlim` - Fluxlimit calculation routines

Fluxlimit routine

Contains python translation of Karl Gebhardt

vdrp.calc_fluxlim.**calc_fluxlim**(*args*, *workdir*)

Equivalent of the rflim0 script and of mklistfl and the rspfl3f scripts.

Calculate the flux limit for a given night and shot, looping over a (by default) 70 x 70 arcsecond grid

Parameters

args [struct] The arguments structure

vdrp.calc_fluxlim.**calc_fluxlim_entrypoint**()

vdrp.calc_fluxlim.**getDefaults**()

vdrp.calc_fluxlim.**get_arguments**(*parser*)

Add command line arguments for the photometry routines, this function can be called from another tool.

Parameters

parser [argparse.ArgumentParser]

vdrp.calc_fluxlim.**main**(*jobnum*, *args*)
Main function.

vdrp.calc_fluxlim.**parseArgs**(*argv*)
Parses configuration file and command line arguments. Command line arguments overwrite configuration file settiongs which in turn overwrite default values.

Args: args (argparse.Namespace): Return the populated namespace.

vdrp.calc_fluxlim.**update_im3d_header**(*ra*, *dec*, *wdir*)
Add header keywords to the image3d.fits

2.2.3 `cltools` - Commandine tools

Interface to astrometry command line tools

Module provide interface routines to the command line tools that are use in Karl Gebhardt's astrometry procedure.

vdrp.cltools.**getoff2**(*fnradec*, *fnshuffle_ifustars*, *radius*, *ra_offset*, *dec_offset*, *logging=None*)
Interface to getoff2.

Args: fnradec (string): Filename for detections. fnshuffle_ifustars (string): Filename of catalog of stars to match to. radius (float): Matching radius. ra_offset (float): RA offset to apply to detecions in *fnradec*

before computing offset.

dec_offset (float): Dec offset to apply to detecions in *fnradec* before computeing offset.

logging (module): Pass logging module if. Otherwise output is passed to the screen.

Notes: Creates files *getoff.out*, *getoff2.out* and intermediate file *shout.ifu*.

Retruns: new_ra_offset (float), new_dec_offset (float): New offset after matching (input offset added).

vdrp.cltools.**immosaicv**(*prefixes*, *fplane_file='fplane.txt'*, *logging=None*)
Interface to immosaicv which creates a mosaic give a set of fits files and x y coordinates. Requires infp. This function will prepare the necessary infp file that is read by immosaicv command line tool. Format of the latter is 20180611T054545_015.fits 015 -450.0 -50.0 20180611T054545_022.fits 022 -349.743 250.336 20180611T054545_023.fits 023 -349.798 150.243 20180611T054545_024.fits 024 -350.0 50.0 . . .

Args: prefixes (list): List file name prefixes for the collapsed IFU images. fplane_file (str): Fplane file name. logging (module): Pass logging module if. Otherwise output is passed to the screen.

vdrp.cltools.**imrot**(*fitsfile*, *angle*, *logging=None*)
Interface to imrot. Rotates fits image by given angle.

Notes: Creates file *imrot.fits*.

Args: fitsfile (str): Input fits file name. angle (float): Rotation angle.

2.2.4 `cofes_vis` - Visualization routines

vdrp.cofes_vis.**cofes_4x4_plots**(*prefix=''*, *outfile_name='CoFeS_plots.png'*, *vmin=-15*, *vmax=25*, *logging=None*)
dir is a string containing the directory with the CoFeS files you wish to plot. If its the local directory you can leave it as an empty string

outfile_name is a string with the output file name. This will be placed in the dir directory

the ifu order is 073 083 093 103 074 084 094 104 075 085 095 105 076 086 096 106

vdrp.cofes_vis.**cofes_plots**(*ifunums*, *filename_array*, *outfile_name*, *vmin=-15*, *vmax=25*, *logging=None*)

filename_array is an array-like object that contains the filenames of fits files to plot. The output plot will be the shape of the input array.

outfile is the output file name including the extension, such as out.fits.

vmin and vmax set the stretch for the images. They are in units of counts; Pixels with vmin and below counts are black. Pixels with vmax and above counts are white.

vdrp.cofes_vis.**main**()

2.2.5 containers - Container structures

exception vdrp.containers.**NoShotsException**

Bases: exceptions.Exception

class vdrp.containers.**DithAll**(*ra=0.0*, *dec=0.0*, *ifuslot=''*, *x=0.0*, *y=0.0*, *x_fp=0.0*, *y_fp=0.0*, *filename=''*, *timestamp=''*, *expname=''*)

class vdrp.containers.**DithAllFile**(*filename=None*)

where(*cond*)

class vdrp.containers.**Spectrum**

This class encapsulates the content of a tmp*.dat spectrum file

Attributes

wl [float] Wavelength

cnts [float] Counts of the spectrum

flx [float] Flux of the spectrum

amp_norm [float] Ampliflier normalization

tp_norm [float] Throughput normalization

ftf_norm [float] Fiber to fiber normalization

err_cts [float]

err_cts_local [float]

err_max_flux [float]

read(*fname*)

class vdrp.containers.**StarObservation**(*num=0.0*, *night=-1*, *shot=-1*, *ra=-1*, *dec=-1*, *x=-1*, *y=-1*, *fname=''*, *shotname=''*, *expname=''*, *offset_ra=-1*, *offset_dec=1*)

Data for one spectrum covering a star observation. This corresponds to the data stored in the l1 file with additions from other files

Attributes

num [int] Star number

night [int] Night of the observation

shot [int] Shot of the observation

ra [float] Right Ascension of the fiber center

dec [float] Declination of the fiber center

x [float] Offset of fiber relative to IFU center in x direction

y [float] Offset of fiber relative to IFU center in y direction

full_fname [str] Filename of the multi extension fits file.

shotname [str] NightvShot shot name

expname [str] Name of the exposure.

dist [float] Distance of the fiber from the star position

offset_ra [float] Offset in ra of the fiber from the star position

offset_dec [float] Offset in dec of the fiber from the star position

fname [str] Basename of the fits filenname

ifuslot [str] IFU slot ID

avg [float] Average of the spectrum

avg_norm [float]

avg_error [float] Error of the average of the spectrum

structaz [float] Azimuth of the telescope structure, read from the image header

set_fname(*fname*)
Split the full filename into the base name and the ifuslot

2.2.6 daophot - Daophot helper routines

Interface to daophot routines.

Module provides interface to daophot.

exception vdrp.daophot.**DaophotException**
Bases: exceptions.Exception

class vdrp.daophot.**DAOPHOT_ALS**(*NL*, *NX*, *NY*, *LOWBAD*, *HIGHBAD*, *THRESH*, *AP1*, *PH_ADU*, *RNOISE*, *FRAD*, *data*)
Bases: object

Reads DAOPHOT ALS files.

static read(*als_file*)
Reads daophot als file.

Notes: Creates file *imrot.fits*.

Args: als_file (str): Input file name.

Returns:

(DAOPHOT_ALS): Object containing all the infromation in the als file.

vdrp.daophot.**allstar**(*prefix*, *psf='use.psf'*, *logging=None*)
Interface to allstar.

Notes: Replaces second part of rdsub. Requires allstar.opt and use.psf, PREFIX.ap to be in place.

Args: prefix (str): File name prefix to call daophot phot for. psf (str): File name for PSF model. logging (module): Pass logging module if. Otherwise output is passed to the screen.

vdrp.daophot.**daomaster**(*logging=None*)
Interface to daomaster.

Notes: replaces "rmaster0". Requires 20180611T054545tot.als and all.mch to be in place.

Args: logging (module): Pass logging module if. Otherwise output is passed to the screen.

vdrp.daophot.**daophot_find**(*prefix*, *sigma*, *logging=None*)
Interface to daophot find.

Notes: Replaces second part of rdcoo. Requires daophot.opt to be in place.

Args: prefix (str): File name prefix to call daophot phot for. sigma (float): Daophot phot sigma parameter. logging (module): Pass logging module if.

Otherwise output is passed to the screen.

vdrp.daophot.**daophot_phot**(*prefix*, *logging=None*)
Interface to daophot phot.

Notes: Replaces first part of rdsub. Requires photo.opt to be in place.

Args: prefix (str): File name prefix to call daophot phot for. logging (module): Pass logging module if. Otherwise output is passed to the screen.

vdrp.daophot.**filter_daophot_out**(*file_in*, *file_out*, *xmin*, *xmax*, *ymin*, *ymax*)
Filter daophot coo ouput files. For close-to-edge detections.

Read the daophot *.coo output file and rejects detections that fall outside xmin - xmax and ymin - ymax. Translated from awk '{s+=1; if (s<=3||($2>4&&$2<45&&$3>4&&$3<45)) print $0}' $1.coo > $1.lst

Notes: Args:

file_in (str): Input file name. file_out (str): Output file name. xmin (float): Detection x coordinate must be larger than this vaule. xmax (float): Detection x coordinate must be smaller than this vaule. ymin (float): Detection y coordinate must be larger than this vaule. ymax (float): Detection y coordinate must be smaller than this vaule.

vdrp.daophot.**mk_daophot_opt**(*args*)

vdrp.daophot.**test_input_files_exist**(*input_files*)
Takes a list of files names and check if they are in place. Raises DaophotException if not.

Args: input_files (list): List of file names to check.

Raises: DaophotException: If not all given files are present.

2.2.7 `extraction` - Spectrum extraction routines

Spectrum extraction routines

Contains python translation of Karl Gebhardt

vdrp.extraction.**extract_star_spectrum**(*starobs*, *args*, *wl*, *wlr*, *wdir*, *prefix=''*)
Equivalent of the rextsp[1] and parts of the rsp1b scripts.

Extract stellar spectra, creating the tmp*.dat files. If prefix is set, it is prefixed to the tmp*dat file names.

Parameters

starobs [list] List with StarObservation objects.

args [struct] The arguments structure

wdir [str] Name of the work directory

prefix [str (optional)] Optional prefix for the output filenames.

Returns

list List of tmp*dat filenames created.

vdrp.extraction.**get_star_spectrum_data**(*ra*, *dec*, *args*, *nightshot*, *multi_shot=False*, *dithall=None*)

This extracts the data about the different observations of the same star on different ifus.

This is essentially the information stored in the l1 file.

Parameters

ra [float] Right Ascension of the star.

dec [float] Declination of the star.

args [struct] The arguments structure

nightshot [tuple] Tuple of strings with night and shot to search. if None, use all shots containing the given RA /DEC

vdrp.extraction.**get_structaz**(*starobs*, *path*)

Equivalent of the rgetadc script Read the STRUCTAZ parameter from the multi extension fits files and fill in the StarObservation entries.

2.2.8 file_tools - File access routines

vdrp.file_tools.**get_dithall_file**(*basedir*, *night*, *shot*)

vdrp.file_tools.**get_mulitfits_file**(*basedir*, *night*, *shot*, *expname*, *fname*)

vdrp.file_tools.**get_norm_file**(*path*, *fname*)

vdrp.file_tools.**get_throughput_file**(*path*, *night*, *shot*)

Equivalent of rtp0 script.

Checks if a night/shot specific throughput file exists.

If true, return the filename, otherise the filename for an average throughput file.

Parameters

path [str] Path to the throughput files

shotname [str] Name of the shot

2.2.9 fit_radec - RA/DEC fitting routines

RA /DEC fitting routine, equivalent of rsp3f script

Contains python translation of Karl Gebhardt

vdrp.fit_radec.**fit_radec**(*args*)

Equivalent of the rsp3f script

args [struct] The arguments structure

vdrp.fit_radec.**fitradec_entrypoint**()

vdrp.fit_radec.**getDefaults**()

vdrp.fit_radec.**get_arguments**(*parser*)
Add command line arguments for the photometry routines, this function can be called from another tool.

Parameters

parser [argparse.ArgumentParser]

vdrp.fit_radec.**main**(*jobnum*, *args*)
Main function.

vdrp.fit_radec.**parseArgs**(*argv*)
Parses configuration file and command line arguments. Command line arguments overwrite configuration file settiongs which in turn overwrite default values.

Args: args (argparse.Namespace): Return the populated namespace.

2.2.10 `fplane_client` - FPlane retrieval routines

vdrp.fplane_client.**get_fplane**(*filename*, *datestr=''*, *actpos=True*, *full=False*)
Obtains fplane file from fplane server at MPE.

Args: filename (str) : Filename that the fplane file should be saved to. datestr (str): Datestring of format YYYYMMDD (e.g. 20180611).

vdrp.fplane_client.**main**()

vdrp.fplane_client.**retrieve_fplane**(*night*, *fplane_txt*, *wdir*)
Saves the fplane file to the target directory and names it fplane.txt.

Args: fplane_txt (str) : Either a specific fplane file is specified here, 'DATABASE' is passed,

or a file pattern is provided e.g. fplane_YYYYMMDD.txt. In case of the latter a substring of format YYYYMMDD is expected. The routine will then search for an fplane file of the current date or pick the next older one. E.g. if shot 2080611v017 is to be analysed and fplane files fplane_2080610.txt and fplane_2080615.txt exist, then fplane_2080610.txt will be picked. In the case of DATABASE the fplane file is retrieved from https://luna.mpe.mpg.de/fplane/.

2.2.11 `jobsplitter` - Jobsplitter - slurm setup tool

vdrp.jobsplitter.**create_job_file**(*fname*, *commands*, *n_nodes*, *jobs_per_file*, *jobs_per_node*, *args*)

vdrp.jobsplitter.**getDefaults**()
Get the defaults for the argument parser. Separating this out from the get_arguments routine allows us to use different defaults when using the jobsplitter from within a differen script.

vdrp.jobsplitter.**get_arguments**(*parser*)
Add command line arguments for the jobsplitter, this function can be called from another tool, adding job splitter support.

Parameters

parser [argparse.ArgumentParser]

vdrp.jobsplitter.**main**(*args*)

vdrp.jobsplitter.**n_needed**(*njobs*, *limit*)

vdrp.jobsplitter.**parse_args**(*argv*)
Command line parser

Parameters

argv [list of strings] list to parsed

Returns

namespace: Parsed arguments

vdrp.jobsplitter.**run**()

2.2.12 mphelpers - MPHelpers - Parallel processing routines

class vdrp.mphelpers.**MPPool**(*jobnum*, *num_proc*)
Pool of threads consuming tasks from a queue

add_task(*func*, **args*, ***kargs*)
Add a task to the queue

wait_completion()
Wait for completion of all the tasks in the queue

class vdrp.mphelpers.**MPWorker**(*name*, *tasks*)
Bases: multiprocessing.process.Process

Thread executing tasks from a given tasks queue

run()
Method to be run in sub-process; can be overridden in sub-class

class vdrp.mphelpers.**ThreadPool**(*num_threads*)
Pool of threads consuming tasks from a queue

add_task(*func*, **args*, ***kargs*)
Add a task to the queue

wait_completion()
Wait for completion of all the tasks in the queue

class vdrp.mphelpers.**ThreadShutDownException**

class vdrp.mphelpers.**ThreadWorker**(*name*, *tasks*)
Bases: threading.Thread

Thread executing tasks from a given tasks queue

run()
Method representing the thread's activity.

You may override this method in a subclass. The standard run() method invokes the callable object passed to the object's constructor as the target argument, if any, with sequential and keyword arguments taken from the args and kwargs arguments, respectively.

vdrp.mphelpers.**mp_run**(*func*, *args*, *rargv*, *parser*)

vdrp.mphelpers.**shutdownThread**()

2.2.13 mplog - Mplog - Parallel process logging

class vdrp.mplog.**MultiProcessingHandler**(*name*, *sub_handler=None*)
Bases: logging.Handler

_format_record(*record*)

_receive()

close()
Tidy up any resources used by the handler.

This version removes the handler from an internal map of handlers, _handlers, which is used for handler lookup by name. Subclasses should ensure that this gets called from overridden close() methods.

emit(*record*)
Do whatever it takes to actually log the specified logging record.

This version is intended to be implemented by subclasses and so raises a NotImplementedError.

send(*s*)

setFormatter(*fmt*)
Set the formatter for this handler.

vdrp.mplog.**install_mp_handler**(*logger=None*)
Wraps the handlers in the given Logger with an MultiProcessingHandler. :param logger: whose handlers to wrap. By default, the root logger.

vdrp.mplog.**setup_mp_logging**(*logfile*, *loglevel*)
Setup the logging and prepare it for use with multiprocessing

2.2.14 photometry - Throughput measurement routines

Photometry routine

Contains python translation of Karl Gebhardt

class vdrp.photometry.**ShuffleStar**(*starid=''*, *shotid=''*, *shuffleid=-1*, *ra=-1.0*, *dec=-1.0*, *u=99.0*, *g=99.0*, *r=99.0*, *i=99.0*, *z=99.0*, *catalog='None'*)
Class to store the information about one star from the shuffle output

Attributes

starid [int] ID for the star.

shotid [int] Shot number of the star observation

shuffleid [int] ID of the star in shuffle catalog

ra [float] Right ascension

dec [float] Declination

catalog [str] Catalog name used to find these.

u [float] U-Band magnitude from the shuffle catalog

g [float] G-Band magnitude from the shuffle catalog

r [float] R-Band magnitude from the shuffle catalog

i [float] I-Band magnitude from the shuffle catalog

z [float] Z-Band magnitude from the shuffle catalog

vdrp.photometry.**extract_star_single_shot**(*ra*, *dec*, *starid*, *args*, *dithall=None*)
Equivalent of the rsp1a2b script.

Run the stellar extraction code for a given ra / dec position.

Parameters

ra [float] Right Ascension of the star.

dec [float] Declination of the star.

starid [int] ID to give to the star / position

args [struct] The arguments structure

vdrp.photometry.**getDefaults**()

vdrp.photometry.**get_arguments**(*parser*)
Add command line arguments for the photometry routines, this function can be called from another tool.

Parameters

parser [argparse.ArgumentParser]

vdrp.photometry.**get_g_band_throughput**(*args*)
Measure the throughput in the SDSS g-Band Equivalent of the rgettp0 script

Parameters

args [struct] The arguments structure

vdrp.photometry.**get_sedfits**(*starobs*, *args*, *wdir*, *nomove=False*)
Run quick_fit to generate the SED fits, if available.

If quick_fit cannot be imported, fall back to copying the files from sed_fit_dir

vdrp.photometry.**get_shuffle_stars**(*nightshot*, *args*, *wdir*)
Rerun shuffle and find the all stars for a given night / shot.

Parameters

nightshot [str] Night + shot name to work on.

args [argparse.Namespace] The script parameter namespace

vdrp.photometry.**main**(*jobnum*, *args*)
Main function.

vdrp.photometry.**mk_sed_throughput_curve**(*args*)
Equivalent of the rgett0b script.

Parameters

args [struct] The arguments structure

vdrp.photometry.**parseArgs**(*argv*)
Parses configuration file and command line arguments. Command line arguments overwrite configuration file settiongs which in turn overwrite default values.

Args: args (argparse.Namespace): Return the populated namespace.

vdrp.photometry.**run**()

vdrp.photometry.**run_biwt**(*data*, *outfile*, *wdir*)
Calculate biweight of the supplied data.

Parameters

data [list] List of the data to be run through biwt.

Returns

n, biwt, error

vdrp.photometry.**run_combsed**(*sedlist*, *sigmacut*, *rmscut*, *outfile*, *wdir*, *plotfile=None*)

Parameters

sedlist [list] List of filenames of SED fits

sigmacut [float] Cut value for sigma

rmscut [float] Cut value for rms

outfile [str] Output filename

plotfile [str (optional)] Optional plot output filename

Returns

n, biwt, error

vdrp.photometry.**run_getsdss**(*filename*, *sdss_file*, *wdir*)
Run getsdss on filename. Equivalent to rsdss file.

Parameters

filename [str] Filename with spectral data

sdss_file [str] Full path and filename to the sdss g-band filter curve.

Returns

The flux in the g-Band.

vdrp.photometry.**run_shuffle_photometry**(*args*, *wdir*)
Equivalent of the rsetstar script. Find all shuffle stars observed for the night / shot given on the command line, and the loop over all stars ra / dec.

Parameters

args [struct] The arguments structure

2.2.15 programs - FORTRAN program interfaces

vdrp.programs.**call_fit2d**(*ra*, *dec*, *outname*, *wdir*)
Call fit2d. Calculate the 2D spatial fit based on fwhm, fiber locations, and ADC. This convolves the PSF over each fiber, for a given input position. It fits the ampltiude, minimizing to a chi^2.

Requires input files generated by run_fit2d

Parameters

ra [float] Right Ascension of the star.

dec [float] Declination of the star.

outname [str] Output filename.

wdir [str] Name of the work directory

vdrp.programs.**call_fitem**(*wl*, *wdir*)
Call fitem requires input files created by run_fitem

The line fitter. It fits a gauss-hermite. input is fitghsp.in.

Parameters

wl [float] Wavelength

vdrp.programs.**call_fitonevp**(*wave*, *outname*, *wdir*)
Call fitonevp

Requires fitghsp.in created by apply_factor_spline

Parameters

wave [float] Wavelength

outname [str] Output filename

wdir [str] Name of the work directory

vdrp.programs.**call_getnormexp**(*nightshot*, *wdir*)
Call getnormexp. Get fwhm and relative normalizations for the frames.

Parameters

name [str] Observation name

vdrp.programs.**call_imextsp**(*filename*, *ifuslot*, *wl*, *wlw*, *tpavg*, *norm*, *outfile*, *wdir*)
Equivalent of the rextsp script, a wrapper around the imextsp fortran routine.

Extracts the spectrum from the multi fits files and writes the tmp*dat. This also calculates the appropriate photon errors, using counting and sky residual errors. This applies the throughput and fiber to fiber.

Parameters

filename [str] The filename to process

ifuslot [str] The ifuslot name

wl [float] The central extraction wavelength

wlw [float] The width of the extraction window around wl

tpavg [float] Throughput average for the spectrum

norm [str] File with fiber to fiber normaliztion for the spectrum

outfile [str] Name of the output filename

wdir [str] Name of the work directory

vdrp.programs.**call_mkimage**(*ra*, *dec*, *starobs*, *wdir*)
Call mkimage, equivalent of rmkim

Reads the out2d file and creates three images of the emission line data, best fit model and residuals, called im[123].fits.

Parameters

ra [float] Right Ascension of the star.

dec [float] Declination of the star.

starobs [list] List of StarObservation objects for the star

vdrp.programs.**call_mkimage3d**(*wdir*)
Run the mkimage3d command, creating an output file called image3d.fits

vdrp.programs.**call_sumspec**(*starname*, *wdir*)
Call sumpspec. Sums a set of spectra, and then bins to 100AA bins. Used for SED fitting.

Parameters

starname [str] Star name used to create the outputn filename (adds specf.dat)

vdrp.programs.**call_sumsplines**(*nspec*, *wdir*)
Call sumsplines, calculate a straight sum of the spectra in a list, including errors. Expects the spectra to be called tmp101 to tmp100+nspec.

Creates a file called splines.out

Parameters

nspec [int] Number of spectra to read.

wdir [str] Name of the work directory

vdrp.programs.**run_fitradecsp**(*ra*, *dec*, *step*, *nstep*, *w_center*, *w_range*, *ifit1*, *starobs*, *specfiles*, *wdir*)
Setup and call fitradecsp. This creates a file called spec.out

Parameters

starobs [list] List of StarObservation structures one for each fiber

specfiles [list] List of filename of the different spec files

2.2.16 setup_fluxlim - Fluxlimit setup routines

Fluxlimit routine

Contains python translation of Karl Gebhardt

exception vdrp.setup_fluxlim.**NoShotsException**
Bases: exceptions.Exception

vdrp.setup_fluxlim.**getDefaults**()

vdrp.setup_fluxlim.**parseArgs**(*argv*)
Parses configuration file and command line arguments. Command line arguments overwrite configuration file settiongs which in turn overwrite default values.

Args: args (argparse.Namespace): Return the populated namespace.

vdrp.setup_fluxlim.**setup_fluxlim**(*args*, *rargs*)
This is equivalent to the rflim0 and rsetfl scripts.

Determine the input values for the flux limit calculation, create the input file, create the slurm file using the jobsplitter and launch it using sbatch

vdrp.setup_fluxlim.**setup_fluxlim_entrypoint**()
Entrypoint to run the flux limit calculation for one night / shot combination

2.2.17 star_extraction - Stellar Extraction routines

Star Extraction routine. Equivalent of rsp1 script.

Extract star at a given RA/DEC using all shots overlapping with these coordinates.

Contains python translation of Karl Gebhardt

vdrp.star_extraction.**apply_factor_spline**(*factor*, *wdir*)
Equivalent of the rawksp[12] scripts

Apply the factor to the splines.out file. The factor is the number of individual shots the star was observed in.

Parameters

factor [int] The factor to apply.

wdir [str] Name of the work directory

vdrp.star_extraction.**average_spectra**(*specfiles*, *starobs*, *wl*, *wlrange*, *wdir*)
Average all observed spectra and fill in the corresponding entries in the StarObservation class.

This corresponds to the ravgsp0 script

Parameters

specfiles [list] List of spectrum filenames.

starobs [list] List with StarObservation objects.

wl [float] Central wavelength for the averaging.

wlrange [float] Half width of the wavelength range for averaging.

vdrp.star_extraction.**average_spectrum**(*spec*, *wlmin*, *wlmax*)
Corresponds to ravgsp0 script. Calculate the average of the spectrum in the range [wlmin, wlmax]

Parameters

spec [Spectrum] Spectrum class object

wlmin [float] Minimum wavelength of range to average.

wlmax [float] Maximum wavelength of range to average.

Returns

average, normaliztaion and uncertainty, equivalent to the spavg*.dat files.

vdrp.star_extraction.**copy_stardata**(*starname*, *starid*, *wdir*)
Copies the result files from workdir results_dir as done by rspstar.

Parameters

starname [str] Star name to copy over.

starid [int] Star ID to use for the final filename.

results_dir [str] Final directory for results.

vdrp.star_extraction.**extract_star**(*args*)
Equivalent of the rsp1a2b script.

Run the stellar extraction code for a given ra / dec position.

Parameters

args [struct] The arguments structure

vdrp.star_extraction.**getDefaults**()

vdrp.star_extraction.**get_arguments**(*parser*)
Add command line arguments for the photometry routines, this function can be called from another tool.

Parameters

parser [argparse.ArgumentParser]

vdrp.star_extraction.**main**(*jobnum*, *args*)
Main function.

vdrp.star_extraction.**parseArgs**(*argv*)
Parses configuration file and command line arguments. Command line arguments overwrite configuration file settiongs which in turn overwrite default values.

Args: args (argparse.Namespace): Return the populated namespace.

vdrp.star_extraction.**run_fit2d**(*ra*, *dec*, *starobs*, *seeing*, *outname*, *wdir*)
Prepare input files for running fit2d, and run it.

Parameters

ra [float] Right Ascension of the star.

dec [float] Declination of the star.

starobs [list] List with StarObservation objects.

seeing [float] Assumed seeing for the observation.

outname [str] Output filename.

vdrp.star_extraction.**run_fitem**(*wl*, *outname*, *wdir*)
Prepare input file for fitem, and run it.

Parameters

wl [float] Wavelength

outname [str] Base output filename.

vdrp.star_extraction.**run_sumlineserr**(*specfiles*, *wdir*)
Prepare input and run sumlineserr. It sums a set of spectra, and then bins to 100AA bins. Used for SED fitting.

Parameters

specfiles [list] List of spectrum filenames.

vdrp.star_extraction.**star_extract_entrypoint**()

2.2.18 `utils` - VDRP utility routines

Utility functions for virus reductions

This code relies on original software from: Copyright (c) 2011-2016, Astropy Developers Copyright (c) 2012, Free Software Foundation

vdrp.utils.**bindir**()

vdrp.utils.**biweight_bin**(*xv*, *x*, *y*)
Compute the biweight location with a moving window of size "order"

vdrp.utils.**biweight_filter**(*a*, *order*, *ignore_central=3*, *c=6.0*, *M=None*, *func=None*)

vdrp.utils.**biweight_filter2d**(*a*, *Order*, *Ignore_central=(3*, *3)*, *c=6.0*, *M=None*, *func=None*)
Compute the biweight location with a moving window of size "order"

vdrp.utils.**biweight_location**(*a*, *c=6.0*, *M=None*, *axis=None*, *eps=1e-08*)
Copyright (c) 2011-2016, Astropy Developers

Compute the biweight location for an array.

Returns the biweight location for the array elements. The biweight is a robust statistic for determining the central location of a distribution.

The biweight location is given by the following equation

where M is the sample mean or if run iterative the initial guess, and u_i is given by

where MAD is the median absolute deviation.

For more details, see Beers, Flynn, and Gebhardt, 1990, AJ, 100, 32B

Parameters

a [array-like] Input array or object that can be converted to an array.

c [float, optional] Tuning constant for the biweight estimator. Default value is 6.0.

M [float, optional] Initial guess for the biweight location.

axis [tuple, optional] tuple of the integer axis values ot calculate over. Should be sorted.

Returns

——-

biweight_location [float] Returns the biweight location for the array elements.

See also:

median_absolute_deviation, *biweight_midvariance*

Examples

This will generate random variates from a Gaussian distribution and return the biweight location of the distribution:

```
>>> from utils import biweight_location
>>> from numpy.random import randn
>>> randvar = randn(10000)
>>> cbl = biweight_location(randvar)
```

vdrp.utils.**biweight_midvariance**(*a*, *c=15.0*, *M=None*, *axis=None*, *eps=1e-08*, *niter=1*)

Copyright (c) 2011-2016, Astropy Developers

Compute the biweight midvariance for an array.

Returns the biweight midvariance for the array elements. The biweight midvariance is a robust statistic for determining the midvariance (i.e. the standard deviation) of a distribution.

The biweight location is given by the following equation

where is given by

where MAD is the median absolute deviation.

is the number of data for which holds, while the summations are over all i up to n:

This is slightly different than given in the reference below, but results in a value closer to the true midvariance.

The midvariance parameter c is typically 9.0.

For more details, see Beers, Flynn, and Gebhardt, 1990, AJ, 100, 32B

Parameters

a [array-like] Input array or object that can be converted to an array.

c [float] Tuning constant for the biweight estimator. Default value is 9.0.

M [float, optional] Initial guess for the biweight location.

axis [tuple, optional] tuple of the integer axis values ot calculate over. Should be sorted.

Returns

biweight_midvariance [float] Returns the biweight midvariance for the array elements.

See also:

median_absolute_deviation, *biweight_location*

Examples

This will generate random variates from a Gaussian distribution and return the biweight midvariance of the distribution:

```
>>> from utils import biweight_midvariance
>>> from numpy.random import randn
>>> randvar = randn(10000)
>>> scl = biweight_midvariance(randvar)
```

vdrp.utils.**configdir**()

vdrp.utils.**createDir**(*directory*)
Creates a directory. Does not raise an excpetion if the directory already exists.

Args: directory (string): Name for directory to create.

vdrp.utils.**is_outlier**(*points*, *thresh=3.5*)
Copyright (c) 2012, Free Software Foundation

Returns a boolean array with True if points are outliers and False otherwise.

vdrp.utils.**mangle_config_pathname**(*path*)

vdrp.utils.**matrixCheby2D_7**(*x*, *y*)

vdrp.utils.**median_absolute_deviation**(*a*, *axis=None*)
Copyright (c) 2011-2016, Astropy Developers

Compute the median absolute deviation.

Returns the median absolute deviation (MAD) of the array elements. The MAD is defined as `median(abs(a - median(a)))`.

Parameters

a [array-like] Input array or object that can be converted to an array.

axis [tuple, optional] Axis along which the medians are computed. The default (axis=None) is to compute the median along a flattened version of the array.

Returns

median_absolute_deviation [ndarray] A new array holding the result. If the input contains integers, or floats of smaller precision than 64, then the output data-type is float64. Otherwise, the output data-type is the same as that of the input.

See also:

`numpy.median`

Examples

This will generate random variates from a Gaussian distribution and return the median absolute deviation for that distribution:

```
>>> from utils import median_absolute_deviation
>>> from numpy.random import randn
>>> randvar = randn(10000)
>>> mad = median_absolute_deviation(randvar)
```

vdrp.utils.**print_bindir**()

vdrp.utils.**print_conffile**()

vdrp.utils.**print_configdir**()

vdrp.utils.**read_all_mch**(*all_mch*)
Reads all.mch and returns dither information.

Args: all_mch (str): Filename, typically all.mch.

Returns:

(OrdereDict): Dictionary of float tuples, with dither offsets, e.g. {1 : (0.,0.), 2 : (1.27,-0.73), 3 : (1.27,0.73)}

vdrp.utils.**read_radec**(*filename*)
Reads radec.dat file and returns ra,dec,pa.

Args: filename (str): Filename, typically radec.dat or radec2.dat.

Returns: float,float,float: 3 element list with RA, DEC and PA

vdrp.utils.**rm**(*ff*)
Takes a list of files names and deletes them. Does not raise an Exception if a specific file was not in place.

Args: ff (list): List of file names to delete.

vdrp.utils.**setup_logging**(*logger*, *logfile*, *loglevel*)

vdrp.utils.**write_conf_file**(*fname*)

vdrp.utils.**write_radec**(*ra*, *dec*, *pa*, *filename*)
Creates radec.dat-type file and returns ra,dec,pa.

Args: ra (float): Right ascension dec (float): declination pa (float): position angle filename (str): Filename, typically radec.dat or radec2.dat.

2.2.19 vdrp_helpers - VDRP helper routines

class vdrp.vdrp_helpers.**VdrpInfo**(**args*, ***kwargs*)
Bases: collections.OrderedDict

classmethod read(*dir*, *filename='vdrp_info.pickle'*)

save(*dir*, *filename='vdrp_info.pickle'*)

vdrp.vdrp_helpers.**read_data**(*filename*)

vdrp.vdrp_helpers.**run_command**(*cmd*, *input=None*, *wdir=None*)
Run and fortran command sending the optional input string on stdin.

Parameters

cmd [str] The command to be run, must be full path to executable

input [str, optional] Input to be sent to the command through stdin.

vdrp.vdrp_helpers.**save_data**(*d*, *filename*)

2.2.20 `vdrprunner` - VDRP batch runner

CHAPTER 3

About

3.1 Authors

The HETDEX collaboration:

- Jan Snigula <snigula@mpe.mpg.de>
- Maximilian Fabricius <mxhf@mpe.mpg.de>
- Daniel Farrow <dfarrow@mpe.mpg.de>

3.2 Changelog

3.3 TODO

CHAPTER 4

Links

- genindex
- modindex
- search

Python Module Index

v

Index

Symbols

A

B

C

D

E

F

G

I

L

M

N

P

R

S

T

U

V

W