The Swift Reduction Package Users' Manual

by Stefano Covino, 30 Oct 2011, v. ***3.10.2***

Background
The Swift Reduction Package (hereafter SRP) is a packet of tools executable on the command line to perform basic reduction and analysis tasks of optical/NIR astronomical data. There are several tools to organize observations, to manage FITS files, etc.
SRP is not meant to be "yet another" analysis package, our goal is indeed to provide simple tools to solve common problems in our daily working activities.
SRP was originally developed in the context of the *Swift* follow-up activities of the Milan GRB team at the INAF/Brera Astronomical Observatory. The package is designes to be an aid to any researcher to drive further observation of a followed-up GRB counterpart and "swift" can therefore be read simply as "rapid", "agile", etc.
The package is continuously upgraded and improved. Within the limits of our basic project choice (i.e. to provide an as simple as possible tools rather than a very powerful but complex reduction/analysis environment) any help is absolutely welcome.

Some technical comment
This package, written in Python (2.7 is the suggested version), has been widely tested only on PC-Linux and on Mac OS X workstations. You are anyway absolutely free to use, modify, redistribute this package as you like.
Of course, in any case, we decline any responsibility for the use of this package. Given that the sources are available, and the algorithm public, the results are entirely under your own responsibility.

Mailing-List
Due to the nature of the **SRP** project it quite likely to have frequent update and improvements of the various routines as well as a continuous bug fixing (and new bugs are definitely introduced after any feature additions...). Therefore, if you want to be warned each time a new version is delivered, please send an e-mail to the following address: stefano.covino@brera.inaf.it with subject simply **SRP**.

Installation
If you are just updating **SRP** the simplest and suggested solution is to download the package from the PyPI archive with:

sudo easy_install -U --script-dir=/usr/local/bin SRPAstro

provided of course you are connected to the web, and that you want your executable files in "/usr/local/bin".

If you, instead, are installing **SRP** for the first time or maybe you are upgrading to a new **Python** release, it is likely you need to install many different libraries **SRP** relies on.

In principle the command:

sudo easy_install --script-dir=/usr/local/bin SRPAstro

should again do the job. You might also consider to install the package in a virtual python environment if you do not want to interfere with the system python installation.

However, some of the required libraries can (will) require more concerned actions to allow their installation. Indeed, in essentially all cases, browsing the web you can quickly find the solution to any problem.

An alternative and strongly advised procedure is to make a smart use of the various package managers available on many platforms (macports, yum, apt-get, etc.). A possible sequence of operations on Mac OSX is the following:

i) *sudo port -v selfupdate*
ii) *sudo port install python27*
iii) *sudo port select --set python python27*
iv) *sudo port install py27-distribute*
v) *sudo port install py27-numpy*
vi) *sudo port install py27-matplotlib*
vii) *sudo port install py27-scipy*
viii) *sudo port install py27-pil*
ix) *sudo easy_install --script-dir=/usr/local/bin SRPAstro*

while, on other Linux platforms, using yum or apt-get rather than port, an analogous sequence should work.
For instance, on a linux-PC running Fedora:

i) *sudo yum update*
ii) *sudo yum install python*
iii) *sudo yum install python-devel*
iv) *sudo yum install python-setuptools*
v) *sudo yum install numpy*
vi) *sudo yum install python-matplotlib*
vii) *sudo yum install scipy*
viii) *sudo yum install python-imaging*
ix) *sudo easy_install --script-dir=/usr/local/bin SRPAstro*

Finally, do not forget to install these two independent packages widely used by several **SRP** tools (you might use again port, yum, etc.).

1. The ESO-Eclipse package. **Eclipse** is a general purpose reduction system that can very easily be interfaced with regular UNIX commands to run pipelines, etc. The package should run on essentially any UNIX release. The suggested version is 5.0.

2. SExtractor, the well known general use photometric package. **SRP** has been tested with **SExtractor** version 2.0 or later. The suggested version is 2.5 essentially because later versions require quite a large set of libraries to be installed.

Step by step "how to"

This are just examples of what you can do with **SRP**. Please, pay attention that the main emphasis in developing these tools is put in the rapidity and friendly use rather than in getting the very best solution for any possible case. However, experience says that in most cases the results are fully acceptable.

Optical imaging data reduction

Let us assume to have a raw dataset in a directory, now we propose to follow a simple recipe that will allow one to get the final scientific frames with the minimum amount of choices (i.e. the procedure is not completely blind, but almost...).

1. The file list.
   - The first step requires to create a list of the FITS files we want to analyse. This task can be exploited in several different ways. A possible example is: *ls *.fits > filelist.ascii*, you can name the output file as you like. It is also possible to list in the output file, one file per line, FITS files located in different directories.
2. The session name
   - Here you define a common "session name", or a string to prefix most of the files you will create later. This is useful if you have different datasets to reduce and/or analyze in the same directory. The command is **SRPSessionName -n Test**, where of course you can substitute to test any string you like. Please remember that any **SRP** command is executed without parameters will show you a short summary.
3. The FITS keywords
   - This is, probably, the most important task. It is the only one which requires a some level of interaction. Basically, we now choose which are the important FITS keywords to later classify our files (i.e. to separate biases from flat-fields, standards from object frames, etc.). The command is: **SRPKeywords -f myfile.fits**, where "myfile.fits" is a generic FITS file in your dataset supposed to contain all the relevant FITS keywords. Obviously, it is also assumed that your dataset is homogeneous (as it should be, i.e. no data from different instrument/telescope together, no different observing techniques, etc.). Executing the command now you see on the terminal a list of FITS keywords and you can select or not those you want pressing "y" or any other key but "s". Pressing "s" you terminate the selection without having to go to the end of the list. Then, in your directory, there is a text file, usually named "TestKeywords.txt" containing the keywords you selected. Nothing prevents you to further modify the file if you need. For very lazy astronomers there is also the possibility to select a pre-selected keyword list available for some combination of telescopes/

instruments. An example of this possibility could be **SRPKeywords -p VLTFORSIMA**. In this case you are clearly selecting a set of keywords for VLT FORS1 or FORS2 imaging data.

4. The classification
   - Now we are ready to classify our FITS files. The command is **SRPClassify -i filelist.ascii** and the output is a text file where you can find all files listed in "filelist.ascii" with the content of the selected keywords. Unless you select a different output file name the file with the classification is named: "TestClassify.txt", where of course "Test" is the selected session name.
5. The selection
   - This is also a very important step. You have to define criteria to classify the input files. There is not a strict sequence to follow. It depends everything on your needs and fantasy. If you have a "classification file" obtained during the previous step and there is a keyword flagging the bias frames you can select only these frames with the command: **SRPSelect -o _bias.txt -k bias** where "bias" is the keyword to be searched in the "classification file". The output file will be named, in this case, "Test_bias.txt". The command performs a simple string search, selecting all entries (rows) in the classification files where the keyword can be found. A smart use of this facility can allow you to obtain all file lists you need for any subsequent analysis. Of course this command is provided to help any user, however if you have specific skill with one of the many UNIX character manipulation tools (awk, sed, etc.) you can use them as well. For particularly complex situation you can of course edit manually the classification file and create the output files as required.
6. Frame extraction
   - It frequently happens to need to select an area of a frame to be removed. For instance because the boundary pixels are damaged or for any other reason. This step of course can be required everywhere during a reduction stage. The goal can be exploited with **SRPCut -i filelist.ascii -e 10 10 20 20** where "10 10 20 20" are the distance from the original frame borders in pixels. When available, astrometric information (CRPIX1...) are properly updated.
7. Bias creation
   - Assuming you have a list of bias frames you can obtain your final "master bias frame" with **SRPBias -i Test_bias.txt -o _bias.fits** where as usual "Test_bias.txt" is the input file list and the output will be "Test_bias.fits". The master bias frame is obtained by a 5σ-clipped average of the input frames unless you provide different parameters.
8. Imaging flat-field creation
   - Again, assuming you have a list of flat-field frames you can derive your "master flat-field" frame with **SRPFlatImaging -i Test_flat.txt -b Test_bias.fits -o _flat.fits** where "Test_flat.txt" is the input file list, "Test_bias.fits" is the "master bias frame" and "Test_flat.fits" will be the

output frame. The master flat-field is computed by 5σ-clipped average of the input frames unless you provide different parameters.

9. Science frame creation
    - Once you have created your “master bias and flat-field frames” if you have a list of science frames you can apply the correction easily with **SRPScienceFramesImaging -i Test_obj.txt -b Test_bias.fits -f Test_flat.fits** where the meaning of the various parameters should now be trivial. This command also creates an output file list, in this case would be “Test_obj_biasflat.txt” to be used for subsequent analyses. The output files will be named “originalname_biasflat.fits”.
10. Frames alignment
    - If you need you can try to align scientific frames with **SRPAlignImaging -i Test_Di.txt** where again “Test_Di.txt” is the input file list obtained, for instance, by the use of **SRPSelect**. The commands produces output FITS files with the extension “shift” and an output file list named “Test_Dishift.txt”. These files should now be aligned. Of course, being a blind procedure, in case of very noisy frames of with a peculiar lack of bright targets the procedure may easily fail. In addition, it works only with frames all of the same size and with no rotation (i.e. pure translations).
    - For more complicated cases, input frames of different sizes, etc. there is a powerful alternative: **SRPImageMapping -i Test_Di.txt**. This command generates an ouput frame with roto-traslation information with respect a reference frame (the first of the list). This is a relatively long task, but will allow you to then choose a set of objects to be analyzed in all your frames consistently. Output is:
        - filename X_shift Y_shift Rotation_angle X_rotation_centre Y_rotation_centre FWHM Common_area Number_of_matched_stars Comment

      The rototraslation function is:
        - def rotoTrasla ((X,Y), x0=0.0, y0=0.0, alpha=0.0, xrotcent=0.0, yrotcent=0.0):
            - x = X-xrotcent
            - y = Y-yrotcent
            - x n = x 0 + x*math.cos(math.radians(alpha))-y*math.sin(math.radians(alpha))
            - y n = y 0 + x*math.sin(math.radians(alpha))+y*math.cos(math.radians(alpha))
            - NX = xn+xrotcent
            - NY = yn+yrotcent
            - return NX,NY
    - The automatic image mapping tries to find a solution applying different approaches. However there will be for sure cases too difficult to manage. You can try to find a solution "by hand" with **SRPRotoTransla -i inputfits.fits -r reffits.fits**. The solution consists in the roto-traslation pa-

rameters to move the object coordinate of the input fits file to the reference frame of the reference fits file. The coordinate center is at the center of the frames.

- Finally you can create a set of aligned frames, if you applies **SRPImage-Mapping**, with **SRPRTAlignImaging -i inputfiles.txt**. This command allows also to generate exposure maps to be used later.

11. Frames average
    - If you have more frames now aligned by means of one of the steps previously described, you might want to derive an average of all these frames. You can try **SRPAverage -i inputfiles.txt -o outave.fits**, which implies all frames have the same size.
    - Else you can try **SRPAdvAverage -i inputfiles.txt -o outave.fits**, which processes frames of different sizes and can compute a sigma-clipped average. In addition, in conjunction with **SRPRTAlignImaging,** it can manage exposure maps.
12. Frame astrometry
    - Once you have obtained one or more final scientific frames it is usually very important to compute an astrometric solutions for them. You can do that with **SRPAstrometry -i inputframe.fits -o outframe.fits**. The script tries to get information from the file headers, however in a lot of cases you have to provide more reliable information, i.e. frame orientation, pointing, etc.

Imaging data analysis

1. Frames photometry
    - You can carry out a photometric analysis by means of SExtractor. The first step requires to create the parameter files needed by SExtractor to compute the photometry. The command is **SRPPhotParSet -p parset**, where “parset” can be a set adapted to a specific instrument or a generic set of parameter files.
    - Then, if you want to obtain photometry for most of the sources in your frame the command should be like this: **SRPPhotometry -g 5 -s 30000 -i Test_Dishift.txt**, where “Test_Dishift.txt” is the list of files to be processed, “-g 5” is the gain to be used for error estimate, “-s 30000” is the saturation value. The command creates a text output file for each input frame with the extension “_photom.dat”. The format of the output files are “Id X Y RA DEC magap emagap sky fmax mag emag FWHM” where “Id” is the identification code for each studied obiect, “X Y” are the position in pixels on the frame, “RA DEC” are the sky coordinate, if available, of the same object, “magap emagap” are the are the aperture magnitudes in the requested aperture for one second exposure and the 1-б error, “mag emag” the integrated magnitude and 1-б error for one second exposure and “FWHM” the full width at half maximum in pixel for the object. This command is just an interface to SExtractor. Objects too close to the frame boundaries are automatically removed. For more detailed off-line analysis, the command

creates (or uses if already available) a set of parameter files that can be modified according to your needs. Then, it is enough you run again **SRPPhotometry** and the new input parameter set will be used. One more comment: given the need to provide an automatic tool, in case your frame shows a very variable background the automatic star-finding routine and background estimator may very likely fail. Please be aware: automatic pipelines are not a substitute for a skilled brain.

- You can also perform aperture photometry for a selected list of objects with a tool developed as an exercise: **SRPMyPhotometry -i obj_list.txt -f FITS_file**. The input file has a simple format: Id X Y for objects to be measured. And Id X Y Mag eMag for objects to be used as reference. These last objects are analyzed and the magnitudes are used to calibrate the other listed sources. The output is:
  - Id X Y MaxFlux Mag eMag CalibratedMag eCalibratedMag Comment MJD Exptime
- In case no reference stars are supplied you can simply provide a zero point. You can also obtain a list of sources to be used with **SRPMyPhotometry** in a field with **SRPSourceFinder -f frame.fits -e**.
- An interesting tool using **SRPImageMapping** and **SRPMyPhotometry** allows one to perform photometry of the same set of stars in multiple frames independently of dithering and rotation. This tool applies the photometry by **SRPMyPhotometry** to the objects reported in the photometry file. The coordinates of the objects are updated according to the rototraslation parameters output of **SRPImageMapping** and calibrated magnitudes are then reported based on the reference stars possibly added to the configuration file. Finally, files with all the magnitudes, 1σ errors, observation time and exposure length are created for each object photometrized. The syntax is the same as for **SRPMyPhotometry** but the input FITS file is substituted by the output of **SRPImageMapping**: **SRPREMPhotometry -f file_map.txt -i obj_list.txt**.This tools has been developed for a quick reduction of massive REM telescope datasets.

2. Photometry calibration
   - Unless you have derived your photometry with a pre-computed zero-point, it is possible to compute the zero-point for each frame you are studying provided you know the magnitude for at least an object in the field with **SRPZeroPoint -i instrmag.txt -l 1 4 5 6 7 14 -c calib.txt -C 1 2 3 4 5 -t 1.5**. In case you are analyzing standard star frames you can also get calibrated magnitudes with **SRPQuery**.

Optical spectroscopy data reduction

Again let us assume to have a raw dataset in a directory. Steps from 1 to 7 of the optical data reduction are still perfectly adequate.

1. Spectroscopy flat-field creation
   - Assuming you have a list of raw flat-field frames you can derive your “master flat-field” frame with **SRPFlatSpectroscopy -i Test_flat.txt -b**

Test_bias.fits -o _flat.fits where “Test_flat.txt” is the input file list, “Test_bias.fits” is the “master bias frame” and “Test_flat.fits” will be the output frame. The flat-field is obtained exactly as for imaging. However, the spectrum of the flat-field lamp is removed dividing the flat-field frame with a frame created computing an average of the all spectrum raw (dispersion is assumed to be along the horizontal axis). Then an artificial image is created with the same size of the original flat-field and after the division we have the final normalized flat-field suitable for spectroscopy.

Spectroscopy data analysis

1. Air / Vacuum wavelength conversion
   - This is a simple tool to convert air wavelength to/from vacuum wavelength. **SRPAirVacuum -A 6562.801**.
2. DLA , IGM, and line profiles
   - We can also derive the absorption factor due to Dumped Lyman$_\alpha$ systems with **SRPDLA -l 0.4 -n 1e21 -z 1.0** and due to IGM with **SRPIGM -l 1.1 -x 0.9 -t 7.8**. For a general transition and a more accurate modeling you can derive the absorption factor with **SRPLineProfile -n 1e20 -b 1 -l 1300 -t Till_3073**. This function compute a Voigt profile taking into account both Döppler and intrinsic broadening.
3. Solar abundances
   - Element abundances in the Sun can be retrieved i.e. with **SRPSolarAbundance -e Ne**. Data are from Asplund et al. (2009, ARA&A, 47, 481).

Magnitude, flux, frequency, absorption and reddening

1. Magnitude to/from flux conversion
   - It is quite common to have to convert magnitudes to physical units. To do that it is enough to run **SRPMagFlux -b R -m 18.0 0.03**. You will get the effective frequency, flux and 1-б error for magnitude, 1-б error and band reported. The output unit is Jansky or erg s-1cm-2A-1. It is of course possible to derive magnitudes from fluxes with the same command: **SRPMagFlux -b R -j 1e-5 2e-7**.
2. Reddening determination and absorption factors
   - Assuming to know the color excess E_{B-V} it is possible to estimate the amount of reddening at a given wavelength with **SRPDustAbs -g MW -w 0.7**. The wavelength unit is micron and the output is expressed in magnitudes. Extinction curves for the MW, the LMC and the SMC (Pei, 1992, ApJ, 395, 130) and a generic starburst galaxy (Calzetti et al., 2000, ApJ, 533, 682). It is also possible to know the absorption factor due to neutral gas in the X-rays (Morrison & McCammon, 1983, ApJ, 270, 119) with **SRPNhAbs -e 1 -n 1e21**.
3. Flux density and energy-frequency conversion
   - **SRPPLFluxDensity** allows you to compute the flux density at any given energy assuming to have a power-law spectrum and the integrated flux,

i.e. **SRPPLFluxDensity -f 2.4e13 -s 1.7**. The integrated flux is supposed to be expresses in erg/s*cm2, the energy in keV, and the flux density is expressed in Jy. With **SRPEnergyFreqFlux -e 1 -j 1e-6** you can convert energy to frequency or wavelength and vice-versa and flux densities from Jy to erg/s cm^2 Å and viceversa. Energy is expressed in eV, frequency in Hz and wavelength in micron.

Tables and catalogues

1. Catalogue query
   - Given a position in the sky and a radius with the command **SRPQuery -a 01 23 45 -d 23 32 00 -c BS -r 120**. The radius unit is arcmin.
2. Table extraction
   - This command helps you to extract only selected columns from a table for further analyses. It is also possible to skip header rows as in **SRPTabExtract -t table.txt -c '2 3 4 5' -o _output.txt -j 2**.
3. Object extraction
   - You can select a single, or more, objects in a table basing on their coordinates with **SRPGetTabEntry -i table.dat -c 2 3 -C 240.45 312.23 -t 1**. It is possible to work with angular or Cartesian coordinates.
4. Table match
   - This command allows you to find the common objects between two tables (i.e. the **SRPPhotometry** output). You must provide the file names of the table, shifts in both axes, and the maximum tolerance (first match and fine tuning) for object matching: **SRPMatch -r refile.dat -m matchfile.dat -t 5 2.5 -o _outfile.dat**. Same table scale and orientation is assumed. If, on the contrary, you want to find common objects with the same angular coordinates in two tables, you can try **SRPMatchCoord -r refile.dat -m matchfile.dat -t 1 -o _outfile.dat**.

Observation management

1. Target visibility
   - Providing geographic position of the observer, target coordinate and a time reference you can know the azimuth and altitude of the target and the Moon and Sun separation. It is thought as a simple aid for observation planning: **SRPVisibility -o 10:11:12 -20:21:22**. In case you do not provide any input apart from target coordinate the ESO-La Silla site and the present date are assumed.
2. Date conversion
   - It could be useful to quickly convert from/to dates in "regular" format (yyyy/mm/dd hh:mm:ss.dd) to Julian Date of Modified Julian Date. This can be done with **SRPCalendar -v -d 2009/4/8 10:51:00**.
3. Finding-chart creation
   - In order to quickly generate nice-looking (hopefully) finding charts you can use **SRPFindingChart -f fitsfilename**.

Statistics

1. Gaussian distribution
   - Providing mean and standard deviation you can generate an arbitrary number of values following a given Gaussian distribution: **SRPGaussDistrib -m 3 -s 0.1 -n 10**.
2. Gaussian probability
   - You can compute Gaussian 1-tail or 2-tail probability distribution given a value in units of sigma: **SRPGaussProb -s 3 -2**.
3. Chi square surface
   - You can easily derive the increment for the chi square function given some degrees of freedom at a given probability with **SRPChiSqIncrement -p 90 -d 5**. Else, you can compute the probability to have a chisquare higher than the obtained value by chance with **SRPChiSqIncrement -c 10 -d 7**.
4. Sigma-clipped average
   - It is often useful to can evaluate a sigma-clipped average for input data from a file. You can do that with **SRPAverSigmaClipping -i inputtab.txt -d 4 -k 5**.
5. Histogram creation
   - This command takes a column from a file as input and computes an histogram. The output is a table with bin center, bin size, and number of objects in the bin. Histogram parameters as minimum (i.e. 1), maximum (i.e. 10) and bin size (i.e. 0.2) can be chosen as in **SRPHistogram -c 2 -t table.txt -o _output.txt -b 1 10 0.2**.
6. Multi-parametric fits
   - This command is an attempt, far from being perfect in particular considering the error management, to provide a tool to carry out multi-parametric fits and Montecarlo error search. Fits are driven minimizing a χ^2 function, and the minimization algorithm is the popular Nelder-Mead Simplex algorithm using only function calls. It is not the most powerful, but it is easy and probably fully adequate for an “as friendly as possible” tool. The function to be fit must be provided in a file, i.e. *inp.py*, which is nothing more than a regular piece of python code imported by the command. This means that you may in principle write as complex functions as you need. An example is reported below:

```
import math

# pars[0] Normalization
# pars[1] Center
# pars[2] FWHM
# pars[3] Continumm level

# vars[0] First independent variable
# vars[1] Second, if any, independent variable and so on

def myfun (pars, vars):
# function Gaussian + constant
```

```
        y = pars[0]*math.exp(-((pars[1]-vars[0])**2)/(2*pars[2]**2)) + pars[3]
return y
```

- The only requirement is that the function to fit is named *myfun*. Some more examples can be:

```
import math

# pars[0] slope
# pars[1] intercept

# vars[0] First independent variable

def myfun (pars, vars):
# a simple straight line
        y = pars[0]*vars[0] + pars[1]
return y
```

- Or, in case you prefer a smoothed power-law:

```
import math

# pars[0] first power-law index
# pars[1] second power-law index
# pars[2] break “time”
# pars[3] normalization
# pars[4] constant

# vars[0] First independent variable

def myfun (pars, vars):
          y= pars[3]/(math.pow((vars[0]/pars[2]),pars[0]) +
                math.pow((vars[0]/pars[2]),pars[1]))+pars[4]
return y
```

- The table with data must follow a simple but strict format: all the independent variables, dependent variable and error. The error is mandatory. An example is reported below:

```
7230.2 1.13E-17 1.13e-18
7232.8 1.40E-17 1.40e-18
7235.4 1.58E-17 1.58e-18
7238.0 1.68E-17 1.68e-18
7240.5 2.10E-17 2.10e-18
```

- where we have a wavelength and corresponding fluxes and errors. Of course all the suggestions to carry out these operations as keeping numbers of the order of unity, etc. are still fully valid. The format of a table can be modified with **SRPTabExtract**.

- A typical call to **SRPFit** can be: **SRPFit -v -d inputData.dat -g '1 1 1 1' -m inp.py**, where the string reported after the -g parameter is the string of the best guesses for the fitted parameters.
- After having obtained a stable fit you may try to estimate the confidence ranges for the fitted parameters. This task is usually rather tricky and not in general easy to automatize. **SRPFit** implements a Montecarlo search within some provided limits. A typical command could be **SRPFit -v -d inputData.dat -g '1 1 1 1' -m inp.py -o _errData.dat -e 90.0 -i '0.9 1.1 0.8 1.2 0.99 1.01 0.7 1.1'**. The best strategy could be to begin the search close to the best fit parameters and then enlarge the search range appending each iteration result to form a large sample of trial to constrain the error confidence regions.

Afterglow data and cosmology

1. Afterglow flux and frequencies
   - **SRPAftTypSynchrFluxConst(Wind)** allows you to compute the cooling (typical)(self absorption) synchrotron frequency (Hz) in case of constant ISM (wind). You compute the flux (Jy) at the typical synchrotron frequency and the flux for a typical afterglow spectrum at any frequency. From Zhang & Meszaros, IJMPA A19, 2385 (2004) and Hurley, Sari and Djorgovski, in "Compact Stellar X-ray Sources" (Cambridge University Press).
2. Cosmological parameter computation
   - This command allows one to compute the Hubble, angular diameter, comoving and luminosity distances from a minimum to a maximum redshift in any expanding universe. The distance modulus is also derived. Universe parameters, i.e. matter density, cosmological constant, etc. can be provided. **SRPCosmology -v -z 1.5** you get the various distances for the default "concordance" cosmological model.

File format management.

1. DAOPHOT output file conversion
   - This command allows to convert **DAOPHOT** output photometric files to a more readable format. With, for instance, **SRPDao2Sky -f filename.als -v -S** you convert a PSF photometry **DAOPHOT** file to the **ESO-Skycat** format.
2. GAIA-Photom output file conversion
   - This command allows to convert **GAIA-Photom** output photometric files to a more readable format. With, for instance, **SRPGAIA2Sky -f filename.dat -v -S** you convert your photometry to the **ESO-Skycat** format.
3. FITS spectra to ASCII conversion
   - 1D FITS spectra can be converted to ASCII files with **SRPFitsSpectrum2ASCII -f spec.fits**.

FITS files management

1. FITS header management
   - FITS header reading or even writing can be performed with **SRPFitsHeaders -f filefits.fits**.
2. FITS file statistics
   - Statistics (mean, standard deviation, median and maximum value) about FITS files can be computed with **SRPFitsStats -i fitsfile.fits**.
3. File filtering
   - If required it is possible to apply a median filter to a set of FITS frames with **SRPImageFiltering -i filelist.txt**.
4. Conversion to/from WCS coordinates can be carried out with **SRPWCSPixel**.
   - You can compute positions on a frame with known astrometry or convert pixel positions to astronomic positions with **SRPWCSPixel -c 2 3 -t data.txt -w file.fits -s**.
5. FITS extensions
   - Several FITS files are produces with extensions. You can quickly see and extract them with **SRPFitsExtension -i fitsfile.fits -e**.

Miscellanea

1. **SRP** running version
   - You can know the present **SRP** version with **SRPVersion**.

List of commands

1. **SRPAdvAverage**
   - Its purpose is to obtain an average frame from all the input frames.
   - SRPAdvAverage [-v] [-h] -i arg1 -o arg2 [-s arg3] [-x arg4]
     -i Input FITS file list
     -s Sigma-clipping level
     -x Input FITS exposure map file list
     -o Output FITS file

     The exposure maps, if available, allow to compensate areas less exposed.
2. **SRPAftSynchrSpectrumConst**
   - Its purpose is to compute the afterglow synchrotron spectrum in case of constant density ISM.
   - SRPAftSynchrSpectrumConst [-b arg1] [-d arg2] [-e arg3] [-f arg4] [-g arg5] [-h] [-n arg6] [-p arg7] [-t arg8] [-v] [-z arg9]
     -b Epsilon B (0,1).
     -e EPSE Epsilon E (0,1).
     -d DIST Luminosity distance (cm).
     -g Isotropic energy (erg).

-n Particle density (cm^-3).
-f Frequency (Hz).
-p Electron distribution index.
-t Time from burst (days).
-z Source redshift.

Afterglow model for constant density environment.

3. **SRPAftSynchrSpectrumWind**
   - Its purpose is to compute the afterglow synchrotron spectrum in case of wind.
   - SRPAftSynchrSpectrumWind [-a arg1] [-b arg2] [-d arg3] [-e arg4] [-f arg5] [-h] [-g arg6] [-p arg7] [-t arg8] [-v] [-z arg9]
     -a Astar.
     -b Epsilon B (0,1).
     -e Epsilon E (0,1).
     -d Luminosity distance (cm).
     -g Isotropic energy (erg).
     -f Frequency (Hz).
     -p Electron distribution index.
     -t Time from burst (days).
     -z Source redshift.

   Afterglow model for wind shaped environemnt.
4. **SRPAirVacuum**
   - Its purpose is to convert air wavelength to vacuum wavelength and viceversa.
   - SRPAirVacuum -A arg1 / -V arg1 [-h] [-v]
     -A Air wavelength (Angstrom)
     -V Vacuum wavelength (Angstrom)
5. **SRPAlignImaging**
   - Its purpose is to align different frames on a common reference defined by the first frame processed.
   - SRPAlignImaging [-h] -i arg1 [-v]
     -i is the list of FITS images to align

   This routine is a wrapper to the “xcorr2d” ESO-Eclipse command.
6. **SRPAstrometry**
   - Its purpose is to compute an astrometric solution for any image.
   - SRPAstrometry [-d] -i arg1 [-c arg2 arg3] [-h] [-N] -o arg4 [-O] [-p arg5 arg6] [-P arg7 arg8] [-r arg9] [-v] [-x arg10 arg11]
     -d Show starting parameter values.
     -i Input FITS file.
     -o Output FITS file.
     -c Reference for equatorial coordinates.
     -p Reference for pixel coordinates.
     -P Pointing coordinates.
     -x Increment per pixel [e.g. -1.0 1.0] (arcsec/pix).
     -r Rotation angle (deg).
     -f Frame size (arcmin).

-O Use USNO-A2 catalogue.
-N Use 2MASS catalogue.

7. **SRPAverage**
   - Its purpose is to obtain an average frame from all the input frames.
   - SRPAverage [-v] [-h] -i arg1 -o arg2
     -i Input FITS file list
     -o Output FITS file
8. **SRPAverSigmaClipping**
   - Its purpose it to compute a sigma-clipped average for input data.
   - SRPAverSigmaClipping -i arg1 -d arg2 [-e arg3] [-h] [-k arg4] [-v]
     -i File with input data
     -d Column positions for data.
     -e Column positions for data errors.
     -k Sigma-clipping value
9. **SRPBias**
   - Its purpose is to obtain a bias frame by means of a plain average of the input frames.
   - SRPBias [-h] -i arg1 -o arg2 [-s arg3] [-v]
     -i is the ascii file containing the list of FITS files to be processed.
     -m median rather than sigma-clipped average.
     -o is the name for the output BIAS file.
     -s sigmal level (default 5)
     The output BIAS file is obtained by a 5σ-clipped average of the input files.
10. **SRPCalendar**
    - Its purpose is to convert dates from/tom various formats.
    - SRPCalendar [-h] [-v] -d arg1 / -j arg2 / -m arg3 / -n arg4
      -j Julian Date
      -m Modified Julian Date (MJD)
      -d Regular Date (UT) (yyyy/mm/dd hh:mm:ss)
      -n Present date
11. **SRPChiSqIncrement**
    - Its purpose is to compute increment for chi squares.
    - SRPChiSqIncrement [-a arg1] [-c arg2] -d arg3 [-p arg4] [-v]
      -a is the accuracy of the chisquare increment computation
      -c is the resulting chisquare for a fit
      -d is the number of degrees of freedom
      -p is the probability.
      The routine allows one to compute the increment for the chi square having a probability 100-prob% to occur randomly. Typical usage is for deriving uncertainties for multiparametric fits. Typical usage is for deriving uncertainties for multiparametric fits. Alternatively, one can compute the probability to have randomly a higher chisquare than the one obtained in a fit.
12. **SRPClassify**

- Its purpose is to extract information from the FITS headers to be used for subsequent classification.
- SRPClassify [-h] [-v] -i arg1 [-k arg2] [-o arg3]
  -i is a file with a list of FITS file to analyse
  -k is a file with the keyword to read
  -o is the output file.
  The script extracts from a set of FITS files information coded in their headers.

13. SRPCosmology

- Its purpose is to derive cosmological data.
- SRPCosmology [-h] [-hubbleconstant arg1] [--omegalambda arg2] [--omegamatter arg3] -z arg4 [-v]
  --hubbleconstant Hubble Constant
  --omegamatter Omega Matter
  --omegalambda Omega Lambda
  -z Redshift

14. **SRPCut**

- Its purpose is to extraxct subimages from a frame.
- SRPCut -e arg1 arg2 arg3 arg4 [-h] -i arg5 [-v]
  -e indicates the distances in pixels from frame border (leftx, lowy, rightx, upy)
  -i is the input file list.
  The script trims a frame extracting a subframe with the given limits preserving astrometric information.

15. SRPDao2Sky

- Its purpose is to convert photometric files generated by DAOPHOT to other more friendly formats.
- SRPDao2Sky [-h] [-v] [-e arg1] -f arg2 [-S] [-z arg3 arg4]
  -e Exposure time (sec) for frame(s)
  -f Input Daophot (.ap,.als) file
  -S ESO-Skycat output
  -z Zero point and error for photometry

16. SRPDLA

- Its purpose is to derive the absorption factor due to DLA systems.
- SRPDLA -l arg1 [-h] [-v] -n arg2 -z arg3
  -l Observed wavelength (micron)
  -n Nh (>= 0, cm^-2)
  -z DLA system redshift (>= 0)
  DLA modeling performed according to Totani et al. (2006, PASP 58, 485)

17. SRPDustAbs

- Its purpose is to compute the amount of reddening at a given wavelength.
- SRPDustAbs [-c arg1] -g arg2 [-h] [-v] -w arg3
  -c E(B-V) color excess (mag).
  -g Kind of extinction curve.
  -w Wavelength (micron)

Extinction curves for the MW, LMC and SMC galaxies following Pei (1992, ApJ, 395, 130) and starburst galaxies following Calzetti et al. (2000, ApJ, 533, 682)

18. **SRPEnergyFreqFlux**
    - Its purpose is to convert energy to frequency or wavelength and vice-versa.
    - SRPEnergyFreqFlux [-a arg1 / -j arg1] -e arg2 / -f arg2 / -w arg2 [-h] [-v]
        -a Flux density in Erg / cm s A
        -e Energy (eV)
        -f Frequency (Hz)
        -j Flux density in Jy
        -w Wavelength (micron)
19. **SRPFindingChart**
    - Its purpose is to draw nice (hopefully) finding-charts.
    - SRPFindingChart [-c arg1 arg2] -f arg3 [-h] [-l arg4] [-o arg5 arg6] [-r arg7] [-s arg8] [-t arg9] [-v]
        -c Image cuts (min max)
        -f Fits file name
        -l Object label
        -o Object coordinates (RAH:RAM:RAS DECD:DECM:DECS)
        -r Object circle radius (arcsec)
        -t Finding-chart title
        -s Finding-chart size (arcmin)
20. **SRPFit**
    - Its purpose is to carry out multi-parametric fits and Montecarlo error search.
    - SRPFit -d arg1 [-e arg2] -f arg3 -g 'arg4' [-h] [-i 'arg5'] -m arg6 [-n arg7] [-o arg8] [-v]
        -d Table containing data
        -e ERR Error search and confidence level (i.e 90.0)
        -g Guess values for parameters to fit [i.e. '2.2 15.2']
        -i Min,max values for error search [i.e. '0 4 10 20']
        -m File with function to fit (i.e. myfunc.py)
        -n Number of trial for Montecarlo search (default 1000)
        -o Output error file
        -f Output function file
        Perform a multi-parametric fit allowing error serach by means of a Montecarlo run.
21. **SRPFitsExtension**
    - Its purpose is to create independent FITS files from each extension present in the original one.
    - SRPFitsExtension [-e] -i arg1 [-h] [-v]
        -e, --extract Extract extensions
        -i Input FITS file list or single FITS file
        -n Plane number to extract
        Extract extensions in a FITS file

22. **SRPFitsHeaders**
    - Its purpose is to read and/or write FITS headers.
      SRPFitsHeader -f arg1 [-h] [-k arg2/-n arg3 arg4 arg5] [-o arg6] [-v]
        -f is FITS file name or a list of FITS files
        -k select a specific keyword
        -n add a new keyword, value, and commentse
        -o new output file
        If a keyword name is not provided, all header entries are shown.
23. **SRPFitsSpectrum2ASCII**
    - Its purpose is to convert a Fits 1D spectrum to an ASCII file.
    - SRPFitsSpectrum2ASCII -f arg1 [-h] [-v]
        -f Input FITS file list or single FITS file
      Convert a FITS spectrum to an ASCII file
24. **SRPFitsStats**
    - Its purpose is to compute basic statistics for FITS files.
    - SRPFitsStats -i arg1 [-h] [-v]
        -i Input FITS file list or single FITS file
      Returns mean, standard deviation, median and maximum value
25. **SRPFlatImaging**
    - Its purpose is to produce a flat field frame for imaging.
    - SRPFlatImaging -b arg1 [-h] -i arg2 -o arg3 [-v]
      -b is the BIAS/DARK/SKY file to be subtracted
      -i is the list of files to be processes
      -m median rather than sigma-clipped average.
      -o is the output FITS file name
      -s sigmal level (default 5)
      It is obtained by a 5σ positive clipped average of the input files.
26. **SRPFlatSpectroscopy**
    - Its purpose is to produce a flat field frame for spectroscopy. SRPFlat-Spectroscopy -b arg1 [-h] -i arg2 -o arg3 [-v]
      -b is the BIAS/DARK/SKY file to be subtracted
      -i is the list of files to be processes
      -m median rather than sigma-clipped average.
      -o is the output FITS file name
      -s sigmal level (default 5)
      The flat-field is obtained by a 5σ positive clipped average of the input files. Then the flat field lamp spectrum is removed dividing by the average response on the whole frame along the spatial direction.
27. **SRPGAIA2Sky**
    - Its purpose is to convert photometric files generated by the GAIA-Photom package to other more friendly formats.
    - SRPGAIA2Sky [-h] [-v] [-e arg1] -f arg2 [-S] [-z arg3 arg4]
        -e Exposure time (sec) for frame(s)
        -f Input GAIA photom file
        -S ESO-Skycat output

-z Zero point and error for photometry

28. **SRPGaussDistrib**
    - Its purpose is to generate number following the Gaussian distribution.
    - SRPGaussDistrib -e arg1 [-h] -n arg2 -s arg3 / -a arg 3 arg4
        -e Distribution expectation value
        -n Number of repetitions
        -s Distribution standard deviation
        -a Distribution asymmetric standard deviation (left, right)
29. **SRPGaussProb**
    - Its purpose is to compute probability for Gaussian distributions.
    - SRPGaussProb -s -1/-2 [-v]
        -s Value in sigma units)
        -1 1-tail distribution
        -2 2-tail distribution
30. **SRPGetTabEntry**
    - Its purpose is to find selected objects in a table.
    - SRPGetTabEntry [-a arg1] -c arg2 -C arg3 arg4 arg5 arg6 arg7 [-h] -i arg8 -I arg9 arg10 arg11 arg12 arg13 arg14 [-t arg15] [-v] [-z arg16]
        -i Input table
        -c Column coordinate positions for input table (col1 col2)
        -C Object coordinates (coord1 coord2)
        -t Maximum tolerance for object association (same units as for the coordinates)
        -a Angular distance if set, else Cartesian distance
31. **SRPHistogram**
    - Its purpose is to compute an histogram of input data.
    - SRPHistogram -c arg1 [-h] [-j arg2] -o arg3 -t arg4 [-v]
        -b Bin data [i.e. min max bin_size]
        -c Column for histogram
        -j Number of header lines to skip
        -o Output file
        -t Table containing data to extract
32. **SRPIGM**
    - Its purpose is to derive the absorption factor due to IGM systems.
    - Usage: SRPIGM -b arg1 [-l arg2] [-h] -t arg3 [-v] -x arg4
        -b Lower IGM redshift (>= 0) [default 6.0]
        -l Observed wavelength (micron)
        -t Upper IGM redshift (>= 0)
        -x Neutral hydrogen fraction (0 <= xHI <= 1)
33. **SRPImageFilter**
    - Its purpose is to apply a median filter to a set of images.
    - SRPImageFilter [-h] -i arg1 [-m arg2] [-v]
        -i file of list if files to be processed.
        -m size of median filter.

      The output files are produced applying a median filter of given size.

34. **SRPImageMapping**
    - Its purpose is to derive rototraslation parameters for a set of images.
    - SRPImageMapping [-v] [-h] [-f] -i arg1 [-l arg2] [-m arg3] [-n arg4] [-o] [-p] [-t]
      -f Filter reference object by means of their FWHM
      -i Input FITS file list
      -m Minimum number of stars in common area for matching (default 5)
      -n Number of objects for matching for matching search (default 10)
      -o Save files with object positions
      -t Force pure translation
      -p Integer pixel shift for pure translation
      -r Maximum tolerance (pixel, default 0.0)
      -l Search deepness (default 2, same paramater as for ESO-eclipse peak)
      The script identifies common objects by means of a triangle match.
35. **SRPKeywords**
    - Its purpose is to select which FITS header entries have to be used for classification.
    - SRPKeywords [-h] -f arg / -p arg [-v] arg
      -f is the name of a FITS file to be used as a teplate for FITS keyword selection
      -p is a set of pre-chosen FITS headers for several instrument/telescope combinations.
      If you are not sure, try with any letter and you will be prompted with a list of the available combinations.
36. **SRPLineProfile**
    - Its purpose is to compute line profile for a specific transition.
    - SRPLineProfile -b [arg1] -l arg2 [-h] [-v] -n arg3 -t arg4 -z arg5
      -b Dumping parameter (>= 0), km s^-1)
      -l Observed wavelength (Angstrom)
      -n Column density (>= 0, cm^-2)
      -t Transition
      -z Redshift (>= 0)
      Line profile computed by Voigt function computation.
37. **SRPMagFlux**
    - Its purpose is to convert magnitudes to/from fluxes.
    - SRPMagFlux -b band -f arg1 arg2 / -m arg1 arg2 / -j arg1 arg2 [-h] [-v]
      -b magnitude/flux band
      -f flux and error (Erg/s/cm2/A)
      -j flux and error (Jy)
      -m magnitude and error
38. **SRPMatch**
    - Its purpose is to find common objects between two tables.
    - SRPMatch [-c arg1 arg2 arg3 arg4] [-h] [-j arg5] -m arg6 [-n arg7] -o arg8 -r arg9 [-s arg10 arg11] -t arg12 [-v]

-c are the x,y column numbers for the reference table and the matching table
-j is the number of entries at the beginning of both tables to be skipped.
-m the matching table
-n the character to identify comment lines to skip
-o is the output file
-r reference table
-s the shift for coordinate match
-t is the maximum tolerance for a positive match.

39. **SRPMatchCoord**
    - Its purpose is to find common objects, with the same angular coordinates, in two tables.
    - SRPMatchCoord [-c arg1 arg2 arg3 arg4] [-h] [-j arg5] -m arg6 [-n arg7] -o arg8 -r arg9 -t arg10 [-v]
      -c are the x,y column numbers for the reference table and the matching table
      -j is the number of entries at the beginning of both tables to be skipped.
      -m the matching table
      -n the character to identify comment lines to skip
      -o is the output file
      -r reference table
      -s the shift for coordinate match
      -t is the maximum tolerance for a positive match.
40. **SRPMyPhotometry**
    - Its purpose is to perform aperture photometry for selected source in a frame.
    - SRPMyPhotometry [-a arg1 arg2] [-e arg3] -f arg4 [-g arg5] [-h] [-H arg6, arg7] -i arg8 [-n arg9 [-r arg10 arg11 arg12] [-s arg13] [-t] [-S] [-v] [-z arg14 arg15]
      -a Observation airmass and coefficient
      -f Input FITS file
      -g Gain (e-/ADU) for error estimate in photometry
      -i Input file
      -s Saturation level (ADU) for frame(s)
      -e Exposure time (sec) for frame(s)
      -n Readout noise (e-)
      -S ESO-Skycat output
      -r Radius (pixel) for aperture photometry (r is os)
      -t Do not fit centroid position
      -z Zero point and error for photometry
      -H FITS file header for exposure time and duration
      [default: MJD-OBS, EXPTIME]
41. **SRPNhAbs**
    - Its purpose is to compute the amount of absorption at a given energy.
    - SRPNhAbs -e arg1 [-h] -n arg2 [-v]
      -e Energy (KeV, 0.03-10).

-n Nh (>= 0, cm^-2)
Based on photoelectric absorption cross sections in Morrison & McCammon (1983, ApJ, 270, 119)

42. SRPPhotometry

- Its purpose is to perform aperture photometry for most of the source in the frame.
- SRPPhotometry [-e arg1] [-g arg2] [-h] [-H arg3 arg4] -i arg5 [-r arg6] [-s arg7] [-S] [-v] [-z arg8 arg9]
  - -i Input FITS file list or single FITS file
  - -g Gain (e-/ADU) for error estimate in photometry
  - -s Saturation level (ADU) for frame(s)
  - -e Exposure time (sec) for frame(s)
  - -S ESO-Skycat output
  - -r Radius (pixel) for aperture photometry
  - -z Zero point and error for photometry
  - -H FITS file header for exposure time, duration, airmass and filter [default: MJD-OBS EXPTIME AIRMASS FILTER]

43. SRPPhotParSet

- Its purpose is to create a set of SExtractor parameter files.
- SRPPhotParSet [-h] [-g / -p arg] [-v]
  - -g Generic SExtractor parameter set
  - -p Pre-selected SExtractor parameter sets

44. SRPPLFluxDensity

- Its purpose is to derive flux density known a power-law spectrum and integrated flux.
- SRPPLFluxDensity [-d arg1] [-e arg2 arg3] -f arg4 [-h] [-s arg5] [-v]
  - -d Output flux density (keV)
  - -e Energy limits min max (keV)
  - -f Integrated flux (erg/cm2 s)
  - -s Spectral slope

45. **SRPQuery**

- Its purpose is to extract a region from a catalogue.

SRPQuery -c arg1 arg2 / -f arg3 -C arg4 [-h] [-o arg5] -r arg6 [-S] [-v]

- -c is to input J200 RA and DEC coordinates.
- -C is the acronym of the catalogue to browse.
- -f FITS file to be used for coordinate center.
- -o optional output file.
- -r is the search radius in arcmin. The search is carried out in a cone.
- -S to have an ouput compatible to be shown with the ESO-skycat package.

For several catalogues you need a working internet connection. Other catalogues are local.

46. SRPREMPhotometry

- Its purpose is to perform aperture photometry for selected source in many frames automatically.
- SRPREMPhotometry [-e arg1] [-g arg2] [-h] [-H arg4 arg5] -i arg6 [-n arg7] [-r arg8 arg9 arg10] [-s arg11] [-S] [-t] [-v] [-w] [-z arg9 arg10]
    -f Input FITS file list
    -i Input photometry file
    -g Gain (e-/ADU) for error estimate in photometry
    -H FITS file header for exposure time and duration
        [default: MJD-OBS, EXPTIME]
    -n Readout noise (e-)
    -r Radius (pixel) for aperture photometry
    -s Saturation level (ADU) for frame(s)
    -t Do not fit centroid position
    -w Force re-write of object position files
    -z Zero point and error for photometry

47. SRPRotoTransla

- Its purpose is to apply a roto-translation with parameters provided by the user.
- SRPRotoTransla [-v] [-h] [-f] -i arg1 -p arg2 arg3 arg4 -r arg5
  -f Filter for FWHM value
  -i Input FITS file
  -p Rototraslation parameters (x0,y0,ang [deg])
  -r Reference FITS file

48. SRPRTAlignImaging

- Its purpose is to align different frames on a common reference defined by the first frame processed and basing on roto-traslation parameters.
- SRPRTAlignImaging -i arg1 [-v] [-x]
    -i Input FITS file list
    -x Generate exposure maps

  The exposure maps can then be used to generate average files with compensated exposures.

49. **SRPScienceFramesImaging**

- Its purpose is to apply bias and flat-field correction to a list of frames.
- SRPScienceFramesImaging -b arg1 -f arg2 [-h] -i arg3 [-v]
  -b Input BIAS FITS file or value
  -f Input FLAT FITS file or value
  -i Input science FITS file list

50. **SRPSelect**

- Its purpose is to allow to create list of frames satisfying some criterion.
- SRPSelect [-i arg1] -k arg2 -o arg3 [-v]
  -i passes to the scripts the file with the list
      of FITS file and keyword values as created, for instance, by SRPClassify.
  -k is the keyword to be searched for.
  -o is the output file with results of the selection.

51. **SRPSessionName**
    - Its purpose is to define a new session name.
    - SRPSessionName [-h] -n arg [-v]
      -n allows one to provide a base prefix for many of the files created by other SRP commands.
52. **SRPSolarAbundance**
    - Its purpose is to derive the Solar abundance of various chemical elements.
    - SRPSolarAbundance [-e arg1] [-h] [-v]
      -e is the element to look for, e.g. Fe
      Data are from Asplund et al. (2009, ARA&A, 47, 481)
53. **SRPSourceFinder**
    - Its purpose is to found sources in a frame.
    - SRPSourceFinder -e/-n -f arg1 [-h] [-m arg2] [-S] [-t arg3] [-v]
      -e Eclipse algorithm (default)
      -f FITS file
      -m Minimum number of connected pixel (for native only)
      -n Native algorithm
      -S Skycat output
      -t Threshold for pixel selection
54. **SRPTabExtract**
    - Its purpose is to extract selected columns from a table.
    - SRPTabExtract -c 'arg1' [-h] [-j arg2] -o arg3 -t arg4 [-v]
      -c Columns for columns [i.e. '2 3 1 2']
      -j Number of header lines to jump
      -o Output file
      -t TABLE Table containing data to extract
55. **SRPVersion**
    - its purpose is to show the running **SRP** version.
    - SRPVersion [-h] [-v]
56. **SRPVisibility**
    - Its purpose is to compute the visibility of a sky object.
    - SRPVisibility [-h] [-l arg1 arg2 / -s arg3] -o arg4 [-t arg5] [-v]
      -l Coordinate location of observing site (dd:mm:ss or dd.dddd)
      -s Observing site
      -o Object coordinates (hh:mm:ss dd:mm:ss or hh.ddd dd.ddd)
      -t Computation time ('yyy/mm/dd hh:mm:ss')
57. **SRPWCSPixel**
    - Its purpose is to convert coordinates to pixel on a specific frame.
    - SRPWCSPixel -c arg1 arg2 [-d] [-h] [-j arg3] -t arg4 [-s] [-v] -w arg5
      -c Columns for coordinates [i.e. 2 3]
      -d Decimal degree data in input [i.e. 152.54166 -10.16944]
      -j Number of header lines to jump
      -s Sexagesimal data in input [i.e. 10:10:10 -10:10:10]
      -t Table containing data to convert

-w FITS file with WCS solution

58. **SRPZeroPoint**
    - Its purpose is to compute magnitude zero-point with instrumental and catalogue data.
    - SRPZeroPoint [-a arg1] -c arg2 -C arg3 arg4 arg5 arg6 arg7 [-h] -i arg8 -I arg9 arg10 arg11 arg12 arg13 arg14 [-t arg15] [-v] [-z arg16]
        -a Extinction coefficient (mag/airmass)
        -c File with catalogue magnitudes
        -C Column positions for Id RA DEC Mag eMag
        -i File with instrumental magnitudes (at 1s)
        -I Column positions for Id RA DEC Mag eMag Airmass
        -t Maximum tolerance for object association (arcsec)
        -z Zero-point for instrumental magnitudes

Bugs, comments, etc.

Of course, as already stated, any contribution from anyone is welcome. In case you find bugs, have improvements to suggest, would like to contribute to the code, etc. Please send an e-mail to Stefano Covino, stefano.covino@brera.inaf.it. We can not promise to take into account all your comments, but we will anyway try to improve the package to meet your needs.

Evolution

- From 1.0 to 1.1:
    - Command to convert magnitudes to fluxes and to determine reddening were added. The possibility to create SExtractor files for different instruments was also implemented.
- From 1.1 to 1.2:
    - The command to perform a local catalogue query was implemented. More training steps proposed.
- From 1.2 to 1.3:
    - Aperture photometry is now reported by SRPPhotometry, there is also the possibility to extract subimages from a frame saving the astrometric information.
- From 1.3 to 1.4:
    - Bug correction for aperture photometry. A table matching tool was added.
- From 1.4 to 1.5:
    - The table match now works by a FFT of the input data. The possibility to computer target sky position is implemented. Some minor correction to help data are provided. It is now possible to provide new zero-point to SRPPhotometry. The command SRPMatchCoord finds common entries for object with the same angular coordinates in two tables.
- From 1.5 to 1.6
    - Better management of coordinate matching for SRPMatchCoord. Bias and flat can now be constants in SRPScienceFrameImaging. Better coordinate management and output in SRPVisibility. SRPGaussDistrib added.
- From 1.6 to 1.7
    - Correction to SRPMatch to work with updated numarray library. New parameter set for TNG Dolores imaging frame classification. SRPDao2Sky added. Improved in final match algorithm in SRPMatch. Now it is found the closest companion and not the first within the given tolerance. For SRPMatch and SRPMatchCoord the sequence of reference and matched tables are followed in the output table too.

- From 1.7 to 1.8.0
  - Improvement for SRPMatch allowing the possibility to force the amount of the displacement between the two tables. Minor corrections to SRPDao2Sky. ASIAGO AFOSC and VLT ISAAC imaging keywords. Move from optik to optparse library. SRPTabExtract and SRPHistogram added.
- From 1.8.0 to 1.9.0
  - SRPFlatSpectroscopy and SRPFit added. Minor corrections to SRPMagFlux and SRPTabExtract. New keyword for the TNGDOLORESIMA set. SRPVisibility with Sun altitude.
- From 1.9.0 to 1.9.5
  - Removal of automatic error estimate from SRPFit because it is too much time consuming and not fully reliable. Error search is something intrinsically difficult to automatise in general. Larger number of function calls and evaluations are allowed. Better pair association algorithm for SRPMatchCoord. Calzetti's extinction law added to SRPDustAbs. REM/ROSS photometric parameter set. NTT EMMI, TNG NICS and NOT AFOSC imaging parameter sets added. Additions to VLT FORS spectroscopy keywords. UVOT filters added to SRPMagFlux. A better porting to the cygwin UNIX flavour has been obtained. A bug occurring when not existent directories are reported in the PATH has been corrected. New keywords for VLT ISAAC. Different filename output extension for SRPPhotometry output filenames if “skycat” format is selected. A correction to the algorithm of SRPTabExtract has been applied. Various minor bugs have been corrected.
- From 1.9.5 to 2.0.0
  - Constant density ISM and wind afterglow parameters as in Hurley, Sari, Djorgovski (in "Compact X-ray Stellar Sources", 2003). X-ray absorption as in Morrison & McCammon (1983). SRPNhAbs command. SRPAftTypSynchrConst, SRPAftTypSynchrFreqWind, SRPAftCoolSynchrConst, SRPAftCoolSynchrFreqWind. Data analysis parameter set for the 2.2m Calar Alto telescope with CAFOS and upgrade for the NOT with AFOSC. Coordinates of the NOT and Calar Alto observatories. SRPMatchCoord now reads "hh:mm:ss" and "dd:mm:ss" coordinate format too.
- From 2.0.0 to 2.1.0
  - Minor bug corrections. Danish with DFOSC, VLT with NACO and TNG with NICS imaging parameters added. User's Manual revised. SRPCosmology command added.
- From 2.1.0 to 2.2.0
  - Minor bug corrections. UVOT photometry calibration upgraded. Better data reading in case of high background for SRPDao2Sky. NTT SofI imaging parameters added. SRPCosmology was rewritten. Some improvements to dust absorption by SRPDustAbs computation were developed. Zero-points for magnitude to flux conversion by SRPMagFlux were upgraded. Conversion from coordinates to pixel is now possible with SRPWCS2Pixel. Conversion of the output of the GAIA-Photom package to other formats can be carried out with SRPGAIA2Sky.
- From 2.2.0 to 2.3.0
  - LBT site coordinates. Better management of jump option in SRPMatch and SRPMatchCoord. GEMINI-N coordinates added. Corrections to site coordinates applied. SRPEnergyFreq and SRPPLFluxDensity added.
- From 2.3.0 to 3.0.0
  - New filters. VLTFORSIPOL added. SExtractor photometric parameter file for REMIR added. SRPImageMapping, SRPRotoTrasla, SRPMyPhotometry and SRPREMPhotometry added.
- From 3.0.0 to 3.1.0
  - Improved flexibility of SRPREMphotometry and SRPMyPhotometry commands. Increased execution velocity for SRPImageMapping. SRPRTAlingImaging and SRPAdvAverage commands added.
- From 3.1.0 to 3.2.0

    - Improved rapidity for SRPAdvAverage. Minor improvements to SRPImageMapping and SRPRotoTransla. Better management of objects not in the field of view for SRPREMPhotometry. Installation procedure now much better explained. Better parameter management for SRPEnergyFreq. SRPCalendar added and a few bugs fixed. Better centering algorithm and magnitude computation. SRPMagFlux is improved. SRPDLA added. SRPFit management improved.
- From 3.2.0 to 3.3.0
    - SRPIGM added. SRPShowSpec added. Minor improvements to SRPVisibility, SRPEnergyFreq, SRPGaussDistrib, SRPQuery, SRPMyPhotometry and SRPREMPhotometry. More options for SRPImageMapping. Better algorithm for SRPRTAlignImaging.
- **From 3.3.0 to 3.4.0**
    - Unicode strings in **SRPWCS2Pixel**. Simplified algorithm for **SRPMatch**. Bug correction in **SRPMagFlux**. Upgrade for **python 2.6** and later versions. Various minor upgrade and bug corrections. **SRPGaussProb** added. *X-shooter* parameters added. Minor correction to **SRPVisibility**.
- **From 3.4.0 to 3.5.0**
    - Better sky and zero-point computation with **SRPMyPhotometry**. Bug correction in **SRPAlignImaging**. Possibility to force integer shifts for pure translation and to provide maximum tolerance in **SRPImageMapping**. **SRPAirVacuum**, **SRPGaussProb**, **SRPSourceFinder** and **SRPFindingChart** added. Improvements to **SRPGaussDistrib**.
- **From 3.5.0 to 3.5.1**:
    - Minor bug correction in **SRPAlignImaging**. **2MASS** catalogue and more functionalities added to **SRPQuery**. TNG Dolores spectroscopy parameters added. Exposure maps for **SRPAdvAverage**.
- **From 3.5.1 to 3.6.0**:
    - Sorted output and various improvements to **SRPSourceFinder**. A few minor bugs corrected. Deepness of search selectable in **SRPImageMapping**. **SRPAstrometry** was added. **TNG-LRS** SExtractor parameter files added. **SRPPhotParSet** improved. **SRPQuery** improved and catalogues of Stetson optical standard stars, Astro-wise standard stars and USNO-A2 added. **SRPZeroPoint** added. **nose** python library added to the installation list. **SRPAverSigmaClipping** added. **asciitable** and **ATpy** now required for installation. **SRPTNGPipelineManager** added. **SRPFitsStats** added. **SRPBias**, **SRPFlatImaging**, **SRPScienceFramesImaging** and **SRPCut** improved.
- **From 3.6.0 to 3.7.0**:
    - Better management of cut area for frame binning in **SRPTNGPipelineManager**. New keywords for **SRPTNGManager** and data saved as integer. **SRPGetTabEntry** added. Better magnitude difference averaging algorithm for **SRPZeroPoint**. **SRPREMPipelineManager** added.
- **From 3.7.0 to 3.8.0:**
    - Position on the detector of the selected object in **SRPREMPipelineManager**. Maximum number of log files in SRP pipeline managers. Collection of NIR catalogues (Arnica, Conica, ESO, Isaac, LCO, MSSSO, SAAO, UKIRT) added to **SRPQuery**. Present time in **SRPCalendar**. Bug correction in **SRPFindingChart**. A few bugs corrected in **SRPMyPhotometry**. **SRPPix2WCS** added. Minor bug corrections for **SRPScienceFramesImaging** and **SRPAstrometry**. Improvement to **SRPCosmology**. **SRPAftSynchrSpectrumConst** and **SRPAftSynchrSpectrumWind** rationalized.
- **From 3.8.0 to 3.9.0:**
    - New filters in **SRPMagFlux**. Better output for GRBs in **SRPREMPipelineManager**. Better Voigt profile in **SRPLineProfile**. Better coding for **SRPDustAbs**. Better check for user identity in **SRP** pipelines. AGN optical standard stars catalogue added for **SRPQuery**. Improved AGN photometry for **SRPREMPipelineManager**. Better effective wavelengths for several filters in **SRPMagFlux**. Better frame downloading for **SRPREM-**

PipelineManager. Better management of file download in **SRPREMPipelineManager**. **SRPWCSPixel** simplified. **SRPAverage** improved. **SRPFitsHeader** added.

- **From 3.9.0 to 3.10.0:**
  - Bug correction and new parameter in **SRPMatch**, **SRPMatchCoord** and **SRPKeyword**. New filters in **SRPMagFlux**. **SRPChiSqIncrement** added. New filter data added to **SRPMagFlux**. **SRPSolarAbundance** added. **SRPImageFilter** added. Better parameter management in **SRPBias** and **SRPFlatImaging**. Bug correction for frame weighted sigma-clipped average. Bug corrected in **SRPRotoTransla**. Bug in absolute path file opening corrected. Improvement of **SRPTNGPipelineManager**. **SRPVersion** added. **SRPFitsExtension** added. Possibility to use a Fits file as a reference in **SRPQuery**. Logic and computation corrections to **SRPIGM**.
- **From 3.10.0 to 3.10.2:**
  - Improvements for **SRPFitsHeader**. New bands added to **SRPMagFlux**. **SRPFitsSpectrum2ASCII** added. Bug correction in **SRPAdvAverage**. **SRPEnergyFreq** converted to **SRPEnergyFreqFlux**. *Ellipticity* parameter added to **SRPPhotometry** output.

Credits, thanks, etc.

A lot of people gave some contribution to the **SRP** and among them I want to quote Nino Cucchiara, Paolo D'Avanzo, Luca Di Fabrizio, Dino Fugazza, Auvet Harutyunyan, Nauzet Hernandez, Domenico Impiombato, Gianluca Israel, Daniele Malesani, Emilio Molinari and Ruben Salvaterra. I also thank ESO and TNG since part of this code was developed during visitorships in Garching and La Palma.