astro-lofar

Release 0.1

Emilio Ceccotti

May 30, 2023

CONTENTS

Astro-LoFAR is a Python package to image fits files, extract spectral information, and handle CASA's MS.

Note: This project is under active development.

CHAPTER

ONE

CONTENTS

1.1 Installation

To use Astro-LoFAR, first download the package:

```
$ git clone https://gitlab.com/ececcotti/astro-lofar.git
```

You can install or upgrade Astro-LoFAR from source via

```
$ cd astro-lofar
$ python3 -m pip install -e .
```

Tip: We suggest to install this package in a virtual environment. If you use a conda virtual environment, you still have to use `pip` to install **AstroLoFAR**, so be sure it is the last package to be installed to avoid conflicts with conda packages!

1.1.1 Usage

You can check whether the package has been correctly installed via

```
$ python3 -c "import astrolofar"
```

1.2 Modules

1.2.1 astrolofar.plotfits

Module to plot fits files.

Note: This module is built based on WSClean fits files, i.e. it does not support image cubes (yet). This means you need a fits file for each frequency/polarization.

You can use the module as

```
>>> from astrolofar import plotfits
```

astrolofar.plotfits.**calc_beam**(*fitsname*)

Calculate the beam size in pixel units.

Parameters
fitsname (*str*) – Name of the fits file. You may include the full path if the file is not in the directory where you ran Python.

Returns
Omega – Area of the beam in pixel units.

Return type
float

astrolofar.plotfits.**gen_fits**(*data*, *outname*, *wcs=None*, *freq=None*, *dfreq=1*, *btype='None'*, *comments=['Generated with astrolofar']*)

Generate a simple fits file from input data.

Parameters

- **data** (*array_like*) – Input array of data.
- **outname** (*str*) – Name of the output fits file without extension.
- **wcs** (*WCS object, default: None*) – WCS object to save sky coordinates in the new fits file. You can use *astrolofar.plotfits.read_fits_wcs()* to extract WCS from an existing fits file that contains them. Default is None.

  Warning: The fits file used to extract WCS must be of the same pixel size and scale of the new fits file.

- **freq** (*float, default: None*) – Reference frequency. If default (None) is used, no frequency axis is added.
- **dfreq** (*float, default: 1*) – Frequency increment. It is only activated if *freq* is not None. Default is 1.
- **btype** (*str, default: None*) – Unit of data. Default is None.
- **comments** (*list [str], default: ['Generated with astrolofar']*) – Any comment that you would like to add in the fits file under the COMMENT keyword. Default is 'Generated with astrolofar'.

Returns
outfits – Name of the output fits file (extension is included).

Return type
str

astrolofar.plotfits.**plot_fits**(*fitsname*, *wcs=None*, *c_rms=None*, *c_fitsname=None*, *c_levels=None*, *vrange=('min', 'max')*, *size=(6, 5)*, *trim=(1, 1)*, *cmap='viridis'*, *scalelog10=False*, *cmaplabel=None*, *savefig=None*, *outname='fitsname'*)

Plot a fits file setting different parameters.

Parameters

- **fitsname** (*str*) – Name of the fits file. You may include the full path if the file is not in the directory where you ran Python.
- **wcs** (*WCS object, default: None*) – WCS object to over-plot sky coordinates. You can use *astrolofar.plotfits.read_fits_wcs()* to extract WCS from an existing fits file that contains them. Default is None.

- **c_rms** (*float, default: None*) – Standard deviation or RMS of the residuals/background to set the scale of contours plot.

  Note: This parameters activate contour plotting.

- **c_fitsname** (*str, default: None*) – Name of the fits file from which countours are extracted. It does not have to be the same of *fitsname*. If default (None) is used, `c_fitsname = fitsname`.

  Note: This parameter does something only if `c_rms` is not None.

- **c_levels** (*list, default: None*) – List of levels at which countours are plotted. The actual levels are plotted at `levels * c_rms`. If deafult (None) is used, *c_levels* is *[2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048]*.

  Note: This parameter does something only if `c_rms` is not None.

- **vrange** (*str or tuple ({str, float}, {str, float}), default: ('min', 'max')*) – Data range that the colormap covers. Supported values are:

  1. ('min', 'max') (tuple (str, str)): The default, the colormap covers the full value range of the data.
  2. 'Qperc' or 'Q%' (str): The colormap range is computed with the Q-th percentile (see *astrolofar.plotfits.vrange_percentile()*).
  3. (*A*, *B*) (tuple (float, float)): The colormap ranges between *A* and *B*, with $A > B$.
  4. Any combination of (1) with (3) (for example, you can set `vrange=('min',100)` or `vrange=(-2.5,'max')`, but you can not use `vrange=('10perc',80)`).

- **size** (*tuple (float, float), default: (6, 5)*) – Width, height in inches of the full plot.
- **trim** (*tuple (float, float), default: (1, 1)*) – Fractions along the x and y axis to crop the image about the image center. A value of 1 means no trim is performed. New ranges are assigned to the axis if different values are assigned: $x_{min} = N_{data,x}(1 - t_x)/2$ and $x_{max} = N_{data,x}(1 + t_x)/2$, where t_x is the input trim value (same for *y*).
- **cmap** (*str or Colormap, default: 'viridis'*) – The Colormap instance or Matplotlib colormap name used to color the data.
- **scalelog10** (*bool, default: False*) – Format the colormap in log10.
- **cmaplabel** (*str, default: None (default: fits BUNIT)*) – Label of the colorbar. If None (default) is used, the `BUNIT` keyword of the input fits file is used.

  Warning: The fits file must have the key `BUNIT`. Any WSClean fits file is formatted in the correct way, as well the spectral index map resulting from `astrolofar.spectrum.extract_spix`.

  savefig : {'pdf', 'png'}, default: None Save the plot in PDF or PNG (with 200 dpi) formats (no plot is showed on screen). If None (default) is used, the plot is not saved on disk but showed on screen.

- **outname** (*str, default: fitsname*) – Name of the output image file without extension. Activated only if *savefig* is not None. If `fitsname` (default) is used, the name of the output file will be the same of the input fits file.

Returns

- **image** (AxesImage)
- **colorbar** (Colorbar)

`astrolofar.plotfits.`**read_fits_data**(*fitsname*)

Open an existing fits file to extract data and their units using astropy.io.fits().

Parameters

fitsname (*str*) – Name of the fits file. You may include the full path if the file is not in the directory where you ran Python.

Returns

- **data** (*2D-array*) – Data saved in the fits file, with shape (*X*,`Y`), where *X* and *Y* are the number of pixels along the x and y axis.
- **freq** (*float*) – Frequency or wavelength of the measurement, stored in the fits at the key `CRVAL3`.
- **units** (*str*) – Physical data units, stored in the fits at the key `BUNIT`.

Note: The fits file should have the keys `BUNIT` and `CRVAL3`. Any WSClean fits file is formatted in the correct way, as well the spectral index map resulting from *astrolofar.spectrum.gen_spix_map*.

`astrolofar.plotfits.`**read_fits_wcs**(*fitsname*)

Open an existing fits file to extract WCS (World Coordinate System).

Parameters

fitsname (*str*) – Name of the fits file. You may include the full path if the file is not in the directory where you ran Python.

Return type

WCS object

Example

```
>>> from astrolofar.plotfits import read_fits_wcs
>>> fitsname = "../tests/test.fits"    # fits file name
>>> wcs = read_fits_wcs(fitsname)
>>> wcs.pixel_to_world(10, 50)
<SkyCoord (FK5: equinox=2000.0): (ra, dec) in deg
    (299.91265067, 40.70179643)>
```

Refer to Astropy WCS to more information.

`astrolofar.plotfits.`**regrid**(*fitsname*, *template*, *new_size=(512, 512)*)

Regrid a fits image to different sizes, using a template fits to transfer WCS information.

Parameters

- **fitsname** (*str*) – Name of the fits file. You may include the full path if the file is not in the directory where you ran Python.

- **template** (*str*) – Name of the template fits file. This must have the size and the pixel scale of the wanted regridded image. It is used to set the correct WCS information in the output regridded fits image. Its size must be the same of *new_size* to avoid incorrect WCS projections.
- **new_size** (*tuple (int, int), default: (512, 512)*) – Image size of the output regridded image in pixels. It must th same of the *template* fits file.

Returns
regridded_data – Regridded data.

Return type
array_like

astrolofar.plotfits.**vrange_percentile**(*data*, *q*)

Calculate minimum and maximum value of a q-th percentile of the data about the mean value using numpy.percentile().

Parameters

- **data** (*array_like*) – Input array of data.
- **q** (*float*) – Percentile to compute, which must be in the range 0-100 (inclusive). To have a range similar to DS9, percentiles are calculated with the qi-th "image percentile" $qi = q + (100 - q)/2$.

Returns
range – 2-item tuple `(vmin, vmax)`, where `vmin = numpy.percentile(data, 100-qi)` and `vmax = numpy.percentile(data, qi)`.

Return type
tuple (float, float)

Example

```
>>> import numpy
>>> from astrolofar.plotfits import vrange_percentile
>>> a = numpy.arange(0,201)
>>> vrange_percentile(a, 95)
(5.0, 195.0)
```

Warning: This percentile is different from the "standard" definition of percentile: `numpy.percentile(a, 95)` outputs 190 instead of 195 because it is not centered on the mean value of the pixel distribution.

astrolofar.plotfits.**vrange_percentile_fits**(*fitsname*, *q*)

Use *astrolofar.plotfits.vrange_percentile()* to compute the q-th percentile directly from a fits file.

Parameters

- **fitsname** (*str*) – Name of the fits file. You may include the full path if the file is not in the directory where you ran Python.
- **q** (*float*) – Percentile to compute which must be in the range 0-100 (inclusive). To have a range similar to DS9, percentiles are calculated with the qi-th "image percentile" $qi = q + (100 - q)/2$.

Returns

range – 2-item tuple `(vmin, vmax)`, where `vmin = numpy.percentile(data, 100-qi)` and `vmax = numpy.percentile(data, qi)`.

Return type

tuple (float, float)

See also:

astrolofar.plotfits.vrange_percentile()

1.2.2 astrolofar.spectrum

Module to extract spectral information from fits files.

Note: This module is built based on WSClean fits files, i.e. it does not support image cubes (yet). This means you need a fits file for each frequency/polarization.

You can use the module as

```
>>> from astrolofar import spectrum
```

astrolofar.spectrum.**gen_mask**(*fitsname*, *rms*, *threshold*, *sigma_kernel=(0, 0)*, *theta_kernel=0*, *outname='fitsname'*)

Generate a mask starting from a fits file. The mask has the same pixel scale and size of the input fits file, but WCS info are not stored. The mask is 0 in pixels with brightness lower than `rms * threshold` and 1 in pixels with brightness higher than `rms * threshold`. A 2D Gaussian smoothing kernel can be applied to avoid sharp pixelized features.

Parameters

- **fitsname** (*str*) – Name of the fits file. You may include the full path if the file is not in the directory where you ran Python.
- **rms** (*float*) – Standard deviation or RMS value of the background/noise level. It must be in the same unit of the data stored in the fits file (e.g. mJy/beam).
- **threshold** (*float*) – Set the minimum brightnes included in the mask as `threshold * sigma`.
- **sigma_kernel** (*tuple (float, float), default: (0, 0)*) – Standard deviation of the Gaussian smoothing kernel in (x, y) before rotating by `theta`. Default is (0, 0), which means no smoothing is applied.
- **theta_kernel** (*float, default: 0*) – Rotation angle of the Gaussian smoothing kernel in units of radians. The angle increases counter-clockwise.
- **outname** (*str, default: fitsname*) – Name of the output mask fits file without extension. If `fitsname` (default) is used, the name of the output file will be the same of the input fits file plus `.mask` (i.e. the final name will be `fitsname.mask.fits`).

Returns

- **mask** (*2D-array*) – Mask array saved in outname.
- **outname** (*str*) – Full name (including extension) of the mask fits file.

See also:

astrolofar.plotfits.gen_fits()

astrolofar.spectrum.**gen_spix_map**(*fitslist*, *nterms=1*, *reffreq=None*, *maskfits=None*, *sigma_kernel=(0, 0)*, *theta_kernel=0*, *wsclean=False*, *outname='spix'*)

Extract a pixel-by-pixel spectral index map from a set of fits files. A minimum of 2 files must be given as input. Higher order maps can also be generated if `nterms`>1; in this case, at least 3 files must be given in input. Spectral index (and higher order) map is saved in a fits file with the same pixel scale, size and WCS of the input fits.

Warning: Every fits file must have the same pixel scale and image size. If not, regrid the fits images into the same pixel scale and size.

Parameters

- **fitslist** (*array of str*) – List of fits files from which spectral index (and higher orders) will be extracted. It must contain more than 2 entries.
- **nterms** (*int, default: 1*) – Number of spectral terms for the logaritmic polynomial function. E.g., if *nterms=1* (default) only a spectral index map is extracted by fitting a standard power law; if *nterms=2* also a curvature map is produced, using a second-order logarithmic polynomial.
- **reffreq** (*float, default: None*) – Reference frequency (in MHz) at which spectral indices (or higher orders parameters) are evaluated. If None (default) is used, *reffreq* will be the central frequency of the fits images in *fitslist*.
- **maskfits** (*str, default: None*) – Mask fits file to extract spectral indices from specific pixels. It can be generated by *astrolofar.spectrum.gen_mask()*. The mask must have the same pixel scale and size of the files in *fitslist*. If None (default) is used, spectral indices will be extracted from evry pixel of the image.
- **sigma_kernel** (*tuple (float, float), default: (0, 0)*) – Standard deviation of the Gaussian smoothing kernel in (x, y) before rotating by `theta`. Default is (0, 0), which means no smoothing is applied.
- **theta_kernel** (*float, default: 0*) – Rotation angle of the Gaussian smoothing kernel in units of radians. The angle increases counter-clockwise.
- **wsclean** (*bool, default: False*) – It is only required when the outputs are used in the forced-spectrum fitting of WSClean. When False (dafault) is selected, pixels not selected by the mask assumes NaN values, otherwise they get arbitrary values of -0.7 for the spectral index map and 0 for the higher orders terms, which are the average values for the synchrotron radiation. An extra label *wsclean* is added to the output fits file.
- **outname** (*str, default: 'spix'*) – Name of the output spectral index (and higher order) fits file without extension. A single name must be given: the generated fits file will be a cube with label *SPIX-cube*, where the third axis contain the spectral index map and higher orders (its dimension is the same of *nterms*). If `spix` (default) is used, the output file will be `spix-SPIX-cube.fits`.

Returns

si_map – Array with spectral index (and higher order) values.

Return type

ndarray

See also:

astrolofar.spectrum.gen_mask(), *astrolofar.spectrum.log_pol()*

astrolofar.spectrum.**log_pol**(*x*, **args*)

Evaluate the logarithmic polynomial using the following formula:

$$y = y_0 x^{c_1} \prod_{i=2}^{n} 10^{c_i \log_{10}^i(x)} ,$$

where $y_0 = 10^{c_0}$ is the normalization factor, $\alpha = c_1$ is the spectral index, and c_i are the higher orders coefficients (e.g. $\beta = c_2$ is the curvature). Using y_0 instad of 10^{c_0} allows for the fitting of negative flux densities that could be present in real observations, e.g. because of calibration errors. The order of the logarithmic polynomial is n; e.g., second-order means 3 terms are used, i.e. c_0, c_1, c_2.

Parameters

- **x** (*array_like*) – The bases. If you are using it for spectrum fitting or other astronomycal purpose, it should be the channel frequency divided by a reference frequency: for example, `x = freq/150.` where the reference frequency is 150 MHz.
- ***args** (*float*) – The coefficients c_i. At least two terms must be used, i.e. c_0 and c_1.

Returns

y – The logaritmic polynomial function of x of order n.

Return type

array_like

1.2.3 astrolofar.wsclean_tools

Tools for WSClean output images.

Note: This module is built based on WSClean fits files, i.e. it does not support image cubes (yet). This means you need a fits file for each frequency/polarization.

You can use the module as

```
>>> from astrolofar import wsclean_tools
```

1.2.4 astrolofar.ms_tools

Tools to manipulate MeasurementSet (MS) files.

You can use the module as

```
>>> from astrolofar import ms_tools
```

astrolofar.ms_tools.**calc_sigma_noise**(*SEFD*, *dt*, *df*)

Calculate the per-visibility noise as

$$\sigma = \frac{SEFD}{\sqrt{2 \Delta t \Delta f}} ,$$

where $SEFD$ is the System Equivalent Flux Density (Jy), Δt is the integration time (seconds), and Δf is the channel width (Hz).

Parameters

- **SEFD** (*float*) – The System Equivalent Flux Density in Jy.

- **dt** (*float*) – The time reolution (i.e. integration time) of one visibility in seconds.
- **df** (*float*) – The frequency resolution (i.e. channel width) of one visibility in Hz.

Returns
sigma – The sigma of the per-visibility thermal noise.

Return type
float

PYTHON MODULE INDEX

a

INDEX