aimfast Documentation

Release 0.4.0

Athanaseus Ramaila

Sep 02, 2019

Contents

1 Contents: **3**

1.1 aimfast . . . 3
1.2 Introduction . . . 3
1.3 Fidelity Matrices . . . 3
1.4 Installation . . . 5
1.5 aimfast . . . 7
1.6 License . . . 12
1.7 Contribute . . . 12
1.8 Contact us . . . 12

2 Indices and tables **13**

Python Module Index **15**

Index **17**

An Astronomical Image Fidelity Assessment Tool

CHAPTER 1

Contents:

1.1 aimfast

An Astronomical Image Fidelity Assessment Tool

1.2 Introduction

Image fidelity is a measure of the accuracy of the reconstructed sky brightness distribution. A related metric, dynamic range, is a measure of the degree to which imaging artifacts around strong sources are suppressed, which in turn implies a higher fidelity of the on-source reconstruction. Moreover, the choice of image reconstruction algorithm also affects the correctness of the on-source brightness distribution. For high dynamic ranges with wide bandwidths, algorithms that model the sky spectrum as well as the average intensity can yield more accurate reconstructions.

1.3 Fidelity Matrices

1.3.1 Image dynamic range

Dynamic range is a measure of the degree to which imaging artifacts around strong sources are suppressed, which in turn implies a higher fidelity of the on-source reconstruction. Here we determine it in three ways: Obtaining the quotient of - highest peak flux ($flux_{peak}$) and the absolute of the minimum flux ($flux_{min}$) around the peak in the residual image. - highest peak flux ($flux_{peak}$) and the rms flux ($flux_{local_rms}$) around the peak in the residual image. - highest peak flux ($flux_{peak}$) and the rms flux ($flux_{grobal_rms}$) in the residual image.

$$DR = \frac{flux_{peak}}{|flux_{min}|}(1) DR = \frac{flux_{peak}}{|flux_{local_rms}|}(2) DR = \frac{flux_{peak}}{|flux_{global_rms}|}(3) \tag{1.1}$$

1.3.2 Statistical moments of distribution

The mean and the variance provide information on the location (general value of the residual flux) and variability (spread, dispersion) of a set of numbers, and by doing so, provide some information on the appearance of the distribution of residual flux in the residual image. The mean and variance are calculated as follows respectively

$$MEAN = \frac{1}{n}\sum_{i=1}^{n}(x_i)(4) \tag{1.2}$$

and

$$VARIANCE = \frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})^2(5) \tag{1.3}$$

whereby

$$STD\_DEV = \sqrt{VARIANCE}(6) \tag{1.4}$$

The third and fourth moments are the skewness and kurtosis respectively. The skewness is the measure of the symmetry of the shape and kurtosis is a measure of the flatness or peakness of a distribution. This moments are used to characterize the residual flux after performing calibration and imaging, therefore for ungrouped data, the r-th moment is calculated as follows:

$$m_r = \frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})^r(7) \tag{1.5}$$

The coefficient of skewness, the 3-rd moment, is obtained by

$$SKEWNESS = \frac{m_3}{{m_2}^{\frac{3}{2}}}(8) \tag{1.6}$$

If there is a long tail in the positive direction, skewness will be positive, while if there is a long tail in the negative direction, skewness will be negative.

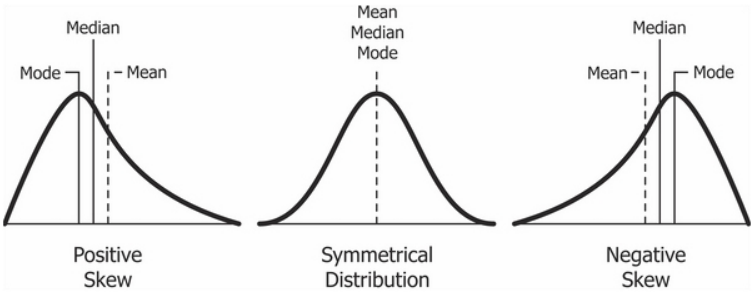

Fig. 1: Figure 1. Skewness of a distribution.

The coefficient kurtosis, the 4-th moment, is obtained by

$$KURTOSIS = \frac{m_4}{{m_2}^2}(9) \tag{1.7}$$

Smaller values (in magnitude) indicate a flatter, more uniform distribution.

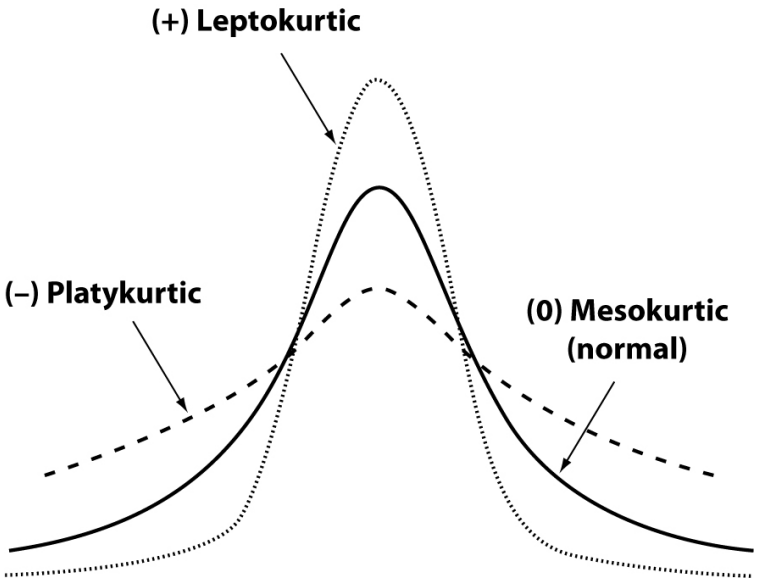

Fig. 2: Figure 2. Kurtosis of a distribution.

1.4 Installation

Installation from source, working directory where source is checked out

```
$ pip install .
```

This package is available on *PYPI*, allowing

```
$ pip install aimfast
```

1.4.1 Command line usage

Get the four (4) statistical moments of the residual image

```
$ aimfast --residual-image cube.residual.fits
```

Get combination of the four (4) moments and dynamic range in one step where argument -af is the multiplying factor of the peak source area:

```
$ aimfast --residual-image cube.residual.fits --restored-image cube.image.fits -af 5
```

or using sky model file (tigger lsm.html or text file):

```
$ aimfast --residual-image cube.residual.fits --tigger-model model.lsm.html -af 5
```

NB: Outputs will be printed on the terminal and dumped into *fidelity_results.json* file. Moreover if the source file names are distinct the output results will be appended to the same json file.

```
$ cat fidelity_results.json
$ {"cube.residual.fits": {"SKEW": 0.124, "KURT": 3.825, "STDDev": 5.5e-05, "MEAN": 4.
↪747e-07},
       "cube.image.fits": {"DR": 53.868}}
```

Additionally, normality testing of the residual image can be performed using the D'Agostino (normaltest) and Shapiro-Wilk (shapiro) analysis, which returns a tuple result, e.g {'NORM': (123.3, 0.1)}, with the z-score and p-value respectively.

```
$ aimfast --residual-image cube.residual.fits --normality-model normaltest
```

Moreover aimfast allows you to swiftly compare two (input-output) tigger models. Currently source flux density and astrometry are examined. It returns an interactive html correlation plots, from which a *.png* file can be easily downloaded or imported to plot.ly.

```
$ aimfast --compare-models model1.lsm.html:model2.lsm.html -af 5 -psf <size_arcsec |␣
↪psf.fits>
```

Where –psf-image | -psf is the Name of the point spread function file or psf size in arcsec. Moreover -as flag can be used to compare all source irrespective of shape (otherwise only point-like source with maj<2" are used).

For Flux density, the more the data points rest on the y=x (or I_out=I_in), the more correlated the two models are.

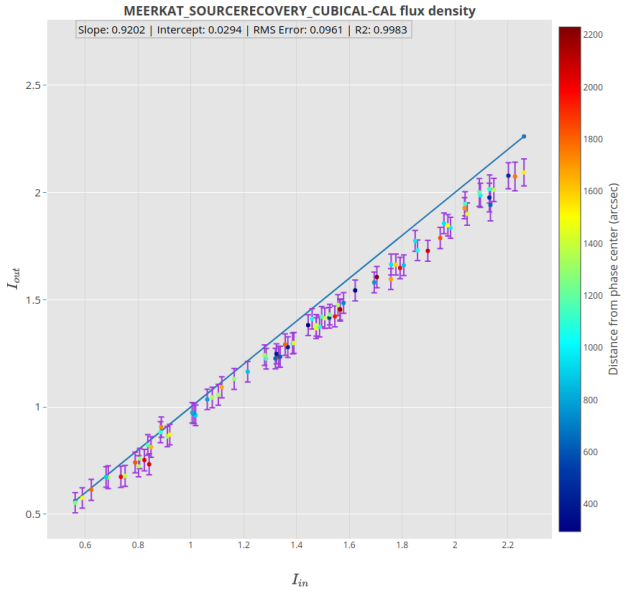

Fig. 3: Figure 3. Input-Output Flux (txt/lsm.html) model comparison

For astrometry, the more sources lie on the y=0 (Delta-position axis) in the left plot and the more points with 1 sigma (blue circle) the more accurate the output source positions.

Furthermore, a comparison of residuals/noise can be performed as follows: To get random residual flux measurements in a *residual1.fits* and *residual2.fits* images

```
$ aimfast --compare-residuals residual1.fits:residual2.fits -dp 100
```

where -dp is the number of data points to sample. To get on source residual flux measurements in a *residual1.fits* and *residual2.fits* images

```
$ aimfast --compare-residuals residual1.fits:residual2.fits --tigger-model model.lsm.
↪html
```

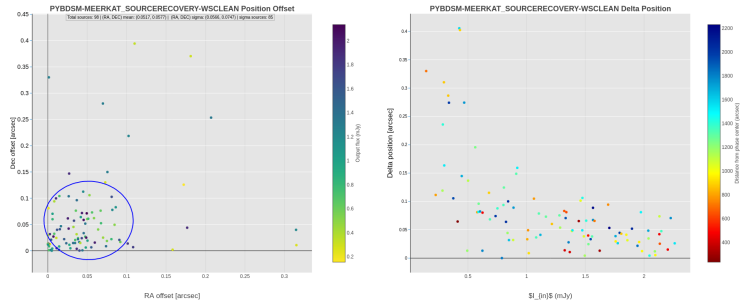

Fig. 4: Figure 4. Input-Output Astrometry (txt/lsm.html) model comparison

where –tigger-model is the name of the tigger model lsm.html file to locate exact source residuals. For random or on source residual noise comparisons, the plot on the left shows the residuals on image 1 and image 2 overlayed and the plot on the right shows the ratios. The colorbar shows the distance of the sources from the phase centre.

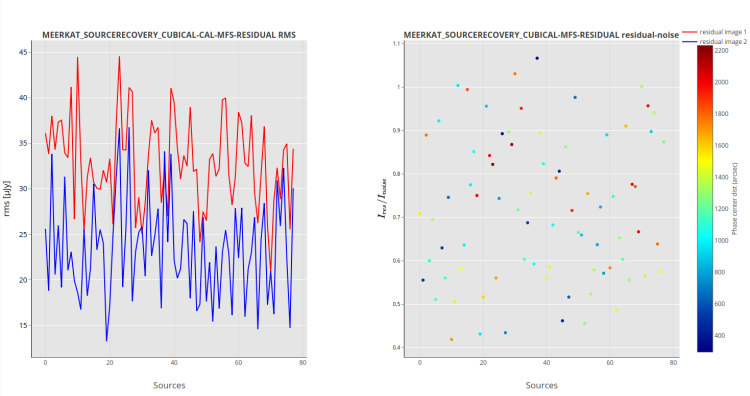

Fig. 5: Figure 5. The random/source residual-to-residual/noise ratio measurements

1.4.2 Jupyter notebook usage

See aimfast in action: NBViewer

1.5 aimfast

1.5.1 aimfast modules

amifast.aimfast module

aimfast.aimfast.**aimfast_plotly**(*X1*, *Y1*, *X2=None*, *Y2=None*, *X3=None*, *Y3=None*, *X4=None*, *Y4=None*, *x_title='x-axis'*, *y_title='y-axis'*, *plot_title='No Ttile'*, *point_labels=None*, *inline=True*, *plot_mode1='markers'*, *plot_mode2='lines'*, *plot_mode3='lines'*, *plot_mode4='lines'*, *xfactor=1*, *yfactor=1*)

Make a simple x-y plot by providing array of Xs and Ys

aimfast.aimfast.**compare_models**(*models*, *tolerance=1e-06*, *plot=True*, *phase_centre=None*, *all_sources=False*)

Plot model1 source properties against that of model2

models [dict] Tigger formatted model files e.g {model1: model2}.

tolerance [float] Tolerace in detecting source from model 2.

plot [bool] Output html plot from which a png can be obtained.

phase_centre [str] Phase centre of catalog (if not already embeded)

all_source: bool Compare all sources in the catalog (else only point-like source)

results [dict] Dictionary of source properties from each model.

aimfast.aimfast.**compare_residuals**(*residuals*, *skymodel=None*, *points=None*, *inline=False*, *area_factor=2.0*)

aimfast.aimfast.**create_logger**()

Create a console logger

aimfast.aimfast.**dec2deg**(*dec_dms*)

Converts declination in dms coordinates to degrees

dec_hms [str] dec in DD:MM:SS format

hms [float] conv_units.radeg: dec in degrees

aimfast.aimfast.**deg2arcsec**(*x*)

Converts 'x' from degrees to arcseconds.

aimfast.aimfast.**fitsInfo**(*fitsname=None*)

Get fits header info.

fitsname [fits file] Restored image (cube)

fitsinfo [dict] Dictionary of fits information e.g. {'wcs': wcs, 'ra': ra, 'dec': dec, 'dra': dra, 'ddec': ddec, 'raPix': raPix, 'decPix': decPix, 'b_size': beam_size, 'numPix': numPix, 'centre': centre, 'skyArea': skyArea}

aimfast.aimfast.**get_aimfast_data**(*filename='fidelity_results.json'*, *dir='.'*)

Extracts data from the json data file

aimfast.aimfast.**get_argparser**()

Get argument parser.

aimfast.aimfast.**get_box**(*wcs*, *radec*, *w*)

Get box of width w around source coordinates radec.

radec [tuple] RA and DEC in degrees.

w [int] Width of box.

wcs [astLib.astWCS.WCS instance] World Coordinate System.

box [tuple] A box centered at radec.

aimfast.aimfast.**get_detected_sources_properties**(*model_1*, *model_2*, *area_factor*, *phase_centre=None*, *all_sources=False*)

Extracts the output simulation sources properties.

models_1 [file] Tigger formatted or txt model 1 file.

models_2 [file] Tigger formatted or txt model 2 file.

area_factor [float] Area factor to multiply the psf size around source.

phase_centre [str] Phase centre of catalog (if not already embeded)

all_source: bool Compare all sources in the catalog (else only point-like source)

(targets_flux, targets_scale, targets_position) [tuple] Tuple of target flux, morphology and astrometry information

aimfast.aimfast.**get_model**(*catalog*)
Get model model object from file catalog

aimfast.aimfast.**get_online_catalog**(*catalog='NVSS', width='1d', thresh=2.0, centre_coord=['0.0', -30.0], catalog_table='nvss_catalog_table.txt'*)
Query an online catalog to compare with local catalog

catalog [str] Name of online catalog to query

width [str] The width of the field iin degrees

thresh [float] Flux density threshold to select sources (mJy)

centre_coord [list] List of central coordinates of the region of interest [RA, DEC]

catalog_table [str] Name of output catalog table with results

aimfast.aimfast.**get_src_scale**(*source_shape*)
Get scale measure of the source in arcsec.

source_shape [lsm object] Source shape object from model

(scale_out_arc_sec, scale_out_err_arc_sec) [tuple] Output source scale with error value

aimfast.aimfast.**image_dynamic_range**(*fitsname*, *residual*, *area_factor=6*)
Gets the dynamic range in a restored image.

fitsname [fits file] Restored image (cube).

residual [fits file] Residual image (cube).

area_factor: int Factor to multiply the beam area.

DR [dict] DRs - dynamic range values.

aimfast.aimfast.**json_dump**(*data_dict*, *root='.'*)
Dumps the computed dictionary into a json file.

data_dict [dict] Dictionary with output results to save.

root [str] Directory to save output json file (default is current directory).

If the fidelity_results.json file exists, it will be append, and only repeated image assessments will be replaced.

aimfast.aimfast.**main**()
Main function.

aimfast.aimfast.**measure_psf**(*psffile*, *arcsec_size=20*)
Measure point spread function after deconvolution.

psfile [fits file] Point spread function file.

arcsec_size [float] Cross section size

r0 [float] Average psf size.

aimfast.aimfast.**model_dynamic_range**(*lsmname*, *fitsname*, *beam_size=5*, *area_factor=2*)
Gets the dynamic range using model lsm and residual fits.

fitsname [fits file] Residual image (cube).

lsmname [lsm.html or .txt file] Model .lsm.html from pybdsm (or .txt converted tigger file).

beam_size [float] Average beam size in arcsec.

area_factor [float] Factor to multiply the beam area.

DR [dict] DRs - dynamic range values.

aimfast.aimfast.**noise_sigma**(*noise_image*)
Determines the noise sigma level in a dirty image with no source

noise_image: file Noise image (cube).

noise_std: float Noise image standard deviation

aimfast.aimfast.**normality_testing**(*data*, *test_normality='normaltest'*, *data_range=None*)
Performs a normality test on the image.

data [numpy.array] Residual residual array.

test_normality [str] Perform normality testing using either *shapiro* or *normaltest*.

data_range [int] Range of data to perform normality testing.

normality [dict] dictionary of stats props. e.g. {'NORM': (123.3, 0.012)} whereby the first element is the statistics (or average if data_range specified) of the datasets and second element is the p-value.

aimfast.aimfast.**plot_astrometry**(*models*, *label=None*, *tolerance=1e-05*, *phase_centre=None*, *all_sources=False*, *dir='.'*)
Plot model-model positions from lsm.html/txt models

models [dict] Tigger/text formatted model files e.g {model1: model2}.

label [str] Use this label instead of the FITS image path when saving data.

tolerance: float Radius around the source to be cross matched.

phase_centre [str] Phase centre of catalog (if not already embeded)

all_source: bool Compare all sources in the catalog (else only point-like source)

results: dict A dict of all the output results

aimfast.aimfast.**plot_morphology**(*models*, *label=None*, *tolerance=1e-05*, *phase_centre=None*, *all_sources=False*, *dir='.'*)
Plot model-model morphology from lsm.html/txt models

models [dict] Tigger/text formatted model files e.g {model1: model2}.

label [str] Use this label instead of the FITS image path when saving data.

tolerance: float Radius around the source to be cross matched.

phase_centre [str] Phase centre of catalog (if not already embeded)

all_source: bool Compare all sources in the catalog (else only point-like source)

results: dict A dict of all the output results

aimfast.aimfast.**plot_photometry**(*models*, *label=None*, *tolerance=1e-05*, *phase_centre=None*, *all_sources=False*, *dir='.'*)

Plot model-model fluxes from lsm.html/txt models

models [dict] Tigger/text formatted model files e.g {model1: model2}.

label [str] Use this label instead of the FITS image path when saving data.

tolerance: float Radius around the source to be cross matched.

phase_centre [str] Phase centre of catalog (if not already embeded)

all_source: bool Compare all sources in the catalog (else only point-like source)

results: dict A dict of all the output results

aimfast.aimfast.**plot_residuals_noise**(*res_noise_images*, *skymodel=None*, *label=None*, *area_factor=2.0*, *points=100*, *dir='.'*)

Plot residual-residual or noise data

res_noise_images: dict Dictionary of residual images to plot {res1.fits: res2.fits}.

skymodel: file Skymodel file to locate on source residuals (lsm.html/txt)

label [str] Use this label instead of the FITS image path when saving data.

area_factor [float] Factor to multiply the beam area.

points: int Number of data point to generate in case of random residuals.

aimfast.aimfast.**plot_spectrum**(*models*, *label=None*, *tolerance=1e-05*, *phase_centre=None*, *all_sources=False*, *dir='.'*)

Plot model-model spectrum from lsm.html/txt models

models [dict] Tigger/text formatted model files e.g {model1: model2}.

label [str] Use this label instead of the FITS image path when saving data.

tolerance: float Radius around the source to be cross matched.

phase_centre [str] Phase centre of catalog (if not already embeded)

all_source: bool Compare all sources in the catalog (else only point-like source)

results: dict A dict of all the output results

aimfast.aimfast.**print_residual_stats**(*residual_images*, *prefix='-'*, *suffix='.fits'*, *replace=''*, *normality='normaltest'*, *units='mJy'*, *dir='.'*)

aimfast.aimfast.**ra2deg**(*ra_hms*)

Converts right ascension in hms coordinates to degrees

ra_hms [str] ra in HH:MM:SS format

hms [float] conv_units.radeg: ra in degrees

aimfast.aimfast.**rad2arcsec**(*x*)

Converts *x* from radians to arcseconds.

aimfast.aimfast.**rad2deg**(*x*)
Converts 'x' from radian to degrees.

aimfast.aimfast.**residual_image_stats**(*fitsname*, *test_normality=None*, *data_range=None*, *threshold=None*, *chans=None*, *mask=None*)
Gets statistcal properties of a residual image.

fitsname [file] Residual image (cube).

test_normality [str] Perform normality testing using either *shapiro* or *normaltest*.

data_range [int, optional] Range of data to perform normality testing.

threshold [float, optional] Cut-off threshold to select channels in a cube

chans [str, optional] Channels to compute stats (e.g. 0~50;100~200)

mask [file] Fits mask to get stats in image

props [dict] Dictionary of stats properties. e.g. {'MEAN': 0.0, 'STDDev': 0.1, 'RMS': 0.1, 'SKEW': 0.2, 'KURT': 0.3, 'MAD': 0.1}.

If normality_test=True, dictionary of stats props becomes e.g. {'MEAN': 0.0, 'STDDev': 0.1, 'SKEW': 0.2, 'KURT': 0.3, 'MAD': 0.1, 'RMS': 0.1, 'NORM': (123.3,0.012)} whereby the first element is the statistics (or average if data_range specified) of the datasets and second element is the p-value.

amifast.tests.test_aimfast module

1.6 License

This project is licensed under the GNU General Public License v3.0 - see license for details.

1.7 Contribute

Contributions are always welcome! Please ensure that you adhere to our coding standards pep8.

1.8 Contact us

Athanaseus Ramaila (aramaila@ska.ac.za)

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

aimfast.aimfast, 7

Index

A

aimfast.aimfast (*module*), 7
aimfast_plotly() (*in module aimfast.aimfast*), 7

C

compare_models() (*in module aimfast.aimfast*), 8
compare_residuals() (*in module aimfast.aimfast*), 8
create_logger() (*in module aimfast.aimfast*), 8

D

dec2deg() (*in module aimfast.aimfast*), 8
deg2arcsec() (*in module aimfast.aimfast*), 8

F

fitsInfo() (*in module aimfast.aimfast*), 8

G

get_aimfast_data() (*in module aimfast.aimfast*), 8
get_argparser() (*in module aimfast.aimfast*), 8
get_box() (*in module aimfast.aimfast*), 8
get_detected_sources_properties() (*in module aimfast.aimfast*), 8
get_model() (*in module aimfast.aimfast*), 9
get_online_catalog() (*in module aimfast.aimfast*), 9
get_src_scale() (*in module aimfast.aimfast*), 9

I

image_dynamic_range() (*in module aimfast.aimfast*), 9

J

json_dump() (*in module aimfast.aimfast*), 9

M

main() (*in module aimfast.aimfast*), 9
measure_psf() (*in module aimfast.aimfast*), 9
model_dynamic_range() (*in module aimfast.aimfast*), 10

N

noise_sigma() (*in module aimfast.aimfast*), 10
normality_testing() (*in module aimfast.aimfast*), 10

P

plot_astrometry() (*in module aimfast.aimfast*), 10
plot_morphology() (*in module aimfast.aimfast*), 10
plot_photometry() (*in module aimfast.aimfast*), 11
plot_residuals_noise() (*in module aimfast.aimfast*), 11
plot_spectrum() (*in module aimfast.aimfast*), 11
print_residual_stats() (*in module aimfast.aimfast*), 11

R

ra2deg() (*in module aimfast.aimfast*), 11
rad2arcsec() (*in module aimfast.aimfast*), 11
rad2deg() (*in module aimfast.aimfast*), 11
residual_image_stats() (*in module aimfast.aimfast*), 12