# imgcube

***Release v0.1***

**Jan 30, 2020**

# Contents:

**1 Installation** **3**
1.1 An `imagecube` Instance . . . . . 3

**Index** **23**

`imgcube` is a set of tools used to manipulate image cubes of molecular emission from protoplanetary disks. This is primarily a collection of functions I routinely use and are not thoroughly tested, so beware!

CHAPTER 1

# Installation

The quickest was to install `imgcube` is to clone the git repository, then, in the source direcrtory,

```
pip install .
```

which should install all the necessary dependancies.

## 1.1 An `imagecube` Instance

**Warning:** This is very much work in progress and documentation may be severely outdated. Use at your own risk.

**class** imgcube.**imagecube**(*fitsfile*, *kelvin=False*, *clip=None*, *resample=1*, *verbose=None*, *preserve_NaN=False*, *suppress_warnings=True*, *center_axes=None*, *center_velocity=None*, *dx0=0.0*, *dy0=0.0*)

**Parameters**

- **fitsfile** (*str/fitsfile object*) – Relative path to the FITS cube or a file object as returned by *astropy.io.fits.open*. The latter allows to first open the file, then add missing entries like pixel size into the header and then pass it to `imagecube`:

```
with fits.open(fname) as hdulist:
    hdulist[0].header['cdelt1'] = -3.405e-06
    hdulist[0].header['cdelt2'] = 3.405e-06
    cube = imagecube(hdulist)
```

- **kelvin** (*Optional[bool/str]*) – Convert the brightness units to [K]. If `True`, use the full Planck law, or if `'RJ'` use the Rayleigh-Jeans approximation. This is not as accurate but does not suffer as much in the low intensity regime.
- **clip** (*Optional[float]*) – Clip the image cube down to a field of view spanning (2 * clip) in [arcsec].

- **resample** (*Optional[int]*) – Resample the data spectrally, averaging over `resample` number of channels.
- **verbose** (*Optional[bool]*) – Print out warning messages messages.
- **suppress_warnings** (*Optional[bool]*) – Suppress warnings from other Python pacakges (for example numpy). If this is selected then `verbose` will be set to `False` unless specified.
- **preserve_NaN** (*Optional[bool]*) – If `False`, convert all `NaN` values to `0.0`.
- **center_axes** (*Optional[float/None]*) – If `None` or `False`, no change to the axes. If `True`, will shift the axes such that 0 is in the center, otherwise a `float` will specify the central offset value. This can be either a tuple, representing the x- and y-axis individually. If just a single value, will apply to both axes.
- **center_velocity** (*Optional[float/None]*) – If `None`, no change is made. If `True`, will center the velocity to 0 [m/s], otherwise a `float` will be the central velocity in [m/s].
- **dx0** (*Optional[float]*) – Recenter the image to this right ascencion offset [arcsec]. This uses 2D interpolation to shift the image relative to the axes.
- **dy0** (*Optional[float]*) – Recenter the image to this declination offset [arcsec]. This uses 2D interpolation to shift the image relative to the axes.

**disk_coords** (*x0=0.0*, *y0=0.0*, *inc=0.0*, *PA=0.0*, *z0=0.0*, *psi=0.0*, *z1=0.0*, *phi=0.0*, *w_i=0.0*, *w_r=1.0*, *w_t=0.0*, *z_func=None*, *w_func=None*, *frame='cylindrical'*)

Get the disk coordinates given certain geometrical parameters and an emission surface. The emission surface is parameterized as a powerlaw profile:

$$z(r) = z_0 \times \left(\frac{r}{1''}\right)^{\psi} + z_1 \times \left(\frac{r}{1''}\right)^{\varphi}$$

Where both `z0` and `z1` are given in [arcsec]. For a razor thin disk, `z0=0.0`, while for a conical disk, as described in Rosenfeld et al. (2013), `psi=1.0`. We can also include a warp which is parameterized by,

$$z_{\rm warp}(r,\, t) = r \times \tan\left(w_i \times \exp\left(-\frac{r^2}{2w_r^2}\right) \times \sin(t - w_t)\right)$$

where `w_i` is the inclination in [radians] describing the warp at the disk center. The width of the warp is given by `w_r` [arcsec] and `w_t` in [radians] is the angle of nodes (where the warp is zero), relative to the position angle of the disk, measured east of north.

**Parameters**

- **x0** (*Optional[float]*) – Source right ascension offset [arcsec].
- **y0** (*Optional[float]*) – Source declination offset [arcsec].
- **inc** (*Optional[float]*) – Source inclination [deg].
- **PA** (*Optional[float]*) – Source position angle [deg]. Measured between north and the red-shifted semi-major axis in an easterly direction.
- **z0** (*Optional[float]*) – Aspect ratio at 1" for the emission surface. To get the far side of the disk, make this number negative.
- **psi** (*Optional[float]*) – Flaring angle for the emission surface.
- **z1** (*Optional[float]*) – Aspect ratio correction term at 1" for the emission surface. Should be opposite sign to `z0`.
- **phi** (*Optional[float]*) – Flaring angle correction term for the emission surface.

- **w_i** (*Optional[float]*) – Warp inclination in [degrees] at the disk center.
- **w_r** (*Optional[float]*) – Scale radius of the warp in [arcsec].
- **w_t** (*Optional[float]*) – Angle of nodes of the warp in [degrees].
- **z_func** (*Optional[callable]*) – User-defined function returning $z$ in [arcsec] at a given radius in [arcsec].
- **w_func** (*Optional[callable]*) – User-defined function returning *z_warp* in [arcsec] at a given radius in [arcsec].
- **frame** (*Optional[str]*) – Frame of reference for the returned coordinates. Either `'polar'` or `'cartesian'`.

**Returns** Disk-frame coordinates. If `frame='cartestian'` this will be three arrays for `(x, y, z)`, otherwise it will be cylindrical coordinates, `(r, theta, z)`.

**Return type** ndarrays

**get_annulus**(*r_min*, *r_max*, *PA_min=None*, *PA_max=None*, *exclude_PA=False*, *x0=0.0*, *y0=0.0*, *inc=0.0*, *PA=0.0*, *z0=0.0*, *psi=1.0*, *z1=0.0*, *phi=1.0*, *w_i=0.0*, *w_r=1.0*, *w_t=0.0*, *z_func=None*, *w_func=None*, *beam_spacing=True*, *return_theta=True*, *as_annulus=True*, *suppress_warnings=True*, *remove_empty=True*, *sort_spectra=True*, ***kwargs*)

Return an annulus (or section of), of spectra and their polar angles. Can select spatially independent pixels within the annulus, however as this is random, each draw will be different.

**Parameters**

- **r_min** (*float*) – Minimum midplane radius of the annulus in [arcsec].
- **r_max** (*float*) – Maximum midplane radius of the annulus in [arcsec].
- **PA_min** (*Optional[float]*) – Minimum polar angle of the segment of the annulus in [degrees]. Note this is the polar angle, not the position angle.
- **PA_max** (*Optional[float]*) – Maximum polar angle of the segment of the annulus in [degrees]. Note this is the polar angle, not the position angle.
- **exclude_PA** (*Optional[bool]*) – If True, exclude the provided polar angle range rather than include.
- **x0** (*Optional[float]*) – Source right ascension offset [arcsec].
- **y0** (*Optional[float]*) – Source declination offset [arcsec].
- **inc** (*Optional[float]*) – Source inclination [deg].
- **PA** (*Optional[float]*) – Source position angle [deg]. Measured between north and the red-shifted semi-major axis in an easterly direction.
- **z0** (*Optional[float]*) – Aspect ratio at 1" for the emission surface. To get the far side of the disk, make this number negative.
- **psi** (*Optional[float]*) – Flaring angle for the emission surface.
- **z1** (*Optional[float]*) – Aspect ratio correction term at 1" for the emission surface. Should be opposite sign to `z0`.
- **phi** (*Optional[float]*) – Flaring angle correction term for the emission surface.
- **w_i** (*Optional[float]*) – Warp inclination in [degrees] at the disk center.
- **w_r** (*Optional[float]*) – Scale radius of the warp in [arcsec].
- **w_t** (*Optional[float]*) – Angle of nodes of the warp in [degrees].

- **z_func** (*Optional[callable]*) – User-defined function returning *z* in [arcsec] at a given radius in [arcsec].
- **w_func** (*Optional[callable]*) – User-defined function returning *z_warp* in [arcsec] at a given radius in [arcsec].
- **beam_spacing** (*Optional[bool/float]*) – If True, randomly sample the annulus such that each pixel is at least a beam FWHM apart. A number can also be used in place of a boolean which will describe the number of beam FWHMs to separate each sample by.
- **annulus** (*Optional[bool]*) – If true, return an annulus instance from `eddy`. Requires `eddy` to be installed.

**Returns** The spectra from each pixel in the annulus. `theta` (ndarray): The midplane polar angles in [radians] of each of the returned spectra. ensemble (annulus instance): An `eddy` annulus instance if `as_ensemble == True`.

**Return type** spectra (ndarray)

**get_vlos**(*r_min*, *r_max*, *PA_min=None*, *PA_max=None*, *exclude_PA=False*, *x0=0.0*, *y0=0.0*, *inc=0.0*, *PA=0.0*, *z0=0.0*, *psi=1.0*, *z1=0.0*, *phi=1.0*, *beam_spacing=True*, *options=None*)

Wrapper for the `get_vlos` function in `eddy.fit_annulus`.

**Parameters**

- **r_min** (*float*) – Minimum midplane radius of the annulus in [arcsec].
- **r_max** (*float*) – Maximum midplane radius of the annulus in [arcsec].
- **PA_min** (*Optional[float]*) – Minimum polar angle of the segment of the annulus in [degrees]. Note this is the polar angle, not the position angle.
- **PA_max** (*Optional[float]*) – Maximum polar angle of the segment of the annulus in [degrees]. Note this is the polar angle, not the position angle.
- **exclude_PA** (*Optional[bool]*) – If True, exclude the provided polar angle range rather than include.
- **x0** (*Optional[float]*) – Source right ascension offset [arcsec].
- **y0** (*Optional[float]*) – Source declination offset [arcsec].
- **inc** (*Optional[float]*) – Source inclination [deg].
- **PA** (*Optional[float]*) – Source position angle [deg]. Measured between north and the red-shifted semi-major axis in an easterly direction.
- **z0** (*Optional[float]*) – Aspect ratio at 1" for the emission surface. To get the far side of the disk, make this number negative.
- **psi** (*Optional[float]*) – Flaring angle for the emission surface.
- **z1** (*Optional[float]*) – Aspect ratio correction term at 1" for the emission surface. Should be opposite sign to z0.
- **phi** (*Optional[float]*) – Flaring angle correction term for the emission surface.
- **beam_spacing** (*Optional[bool/float]*) – If True, randomly sample the annulus such that each pixel is at least a beam FWHM apart. A number can also be used in place of a boolean which will describe the number of beam FWHMs to separate each sample by.
- **options** (*Optional[dict]*) – Dictionary of options for `get_vlos`.

**Returns** Line of sight velocities.

**Return type** array

**sky_to_disk**(*coords*, *frame_in='polar'*, *frame_out='polar'*, *x0=0.0*, *y0=0.0*, *inc=0.0*, *PA=0.0*, *z0=0.0*, *psi=1.0*, *z1=0.0*, *phi=0.0*, *w_i=0.0*, *w_r=1.0*, *w_t=0.0*, *z_func=None*, *w_func=None*)

A convenience function for annotating plots. Convery sky-plane coordinates, either in cartestian or polar coordinates, to disk-plane coordinates based on the provided geometrical properties.

For more information on the geometrical parameters, see the *disk_coords()* documentation.

**Parameters**

- **coords** (*list*) – Midplane coordaintes to transform. If cartesian coordaintes, units of [arcsec, arcsec], or [arcsec, degrees] if in polar.
- **frame_in** (*Optional[str]*) – Frame of input coordinates, either `'cartesian'` or `'polar'`.
- **frame_out** (*Optional[str]*) – Frame of the output coordinates, either `'cartesian'` or `'polar'`.
- **x0** (*Optional[float]*) – Source right ascension offset [arcsec].
- **y0** (*Optional[float]*) – Source declination offset [arcsec].
- **inc** (*Optional[float]*) – Source inclination [deg].
- **PA** (*Optional[float]*) – Source position angle [deg]. Measured between north and the red-shifted semi-major axis in an easterly direction.
- **z0** (*Optional[float]*) – Aspect ratio at 1" for the emission surface. To get the far side of the disk, make this number negative.
- **psi** (*Optional[float]*) – Flaring angle for the emission surface.
- **z1** (*Optional[float]*) – Aspect ratio correction term at 1" for the emission surface. Should be opposite sign to `z0`.
- **phi** (*Optional[float]*) – Flaring angle correction term for the emission surface.
- **w_i** (*Optional[float]*) – Warp inclination in [degrees] at the disk center.
- **w_r** (*Optional[float]*) – Scale radius of the warp in [arcsec].
- **w_t** (*Optional[float]*) – Angle of nodes of the warp in [degrees].
- **z_func** (*Optional[callable]*) – User-defined function returning z in [arcsec] at a given radius in [arcsec].
- **w_func** (*Optional[callable]*) – User-defined function returning z_warp in [arcsec] at a given radius in [arcsec].

**Returns** The disk-frame coordinates in either `(x, y)` or `(r, t)` depending on `frame_out`.

**Return type** array

**disk_to_sky**(*coords*, *frame_in='cylindrical'*, *x0=0.0*, *y0=0.0*, *inc=0.0*, *PA=0.0*, *z0=0.0*, *psi=1.0*, *z1=0.0*, *phi=0.0*, *w_i=0.0*, *w_r=1.0*, *w_t=0.0*, *z_func=None*, *w_func=None*, *return_idx=False*)

For a given disk midplane coordinate, either (r, theta) or (x, y), return interpolated sky coordiantes in (x, y) for plotting. The input needs to be a list like:

```
coords = ([r0, t0], [r1, t1], ..., [rN, tN])
```

If you have an array of values, `rvals` and `tvals` then,

```
coords = np.vstack([rvals, tvals]).T
```

**Parameters**

- **coords** (*list*) – Midplane coordaintes to find in (x, y) in [arcsec, arcsec] or (r, theta) in [arcsec, deg].
- **frame** (*Optional[str]*) – Frame of input coordinates, either 'cartesian' or 'polar'.
- **x0** (*Optional[float]*) – Source right ascension offset [arcsec].
- **y0** (*Optional[float]*) – Source declination offset [arcsec].
- **inc** (*Optional[float]*) – Source inclination [deg].
- **PA** (*Optional[float]*) – Source position angle [deg]. Measured between north and the red-shifted semi-major axis in an easterly direction.
- **z0** (*Optional[float]*) – Aspect ratio at 1" for the emission surface. To get the far side of the disk, make this number negative.
- **psi** (*Optional[float]*) – Flaring angle for the emission surface.
- **z1** (*Optional[float]*) – Aspect ratio correction term at 1" for the emission surface. Should be opposite sign to `z0`.
- **phi** (*Optional[float]*) – Flaring angle correction term for the emission surface.
- **w_i** (*Optional[float]*) – Warp inclination in [degrees] at the disk center.
- **w_r** (*Optional[float]*) – Scale radius of the warp in [arcsec].
- **w_t** (*Optional[float]*) – Angle of nodes of the warp in [degrees].
- **z_func** (*Optional[callable]*) – User-defined function returning z in [arcsec] at a given radius in [arcsec].
- **w_func** (*Optional[callable]*) – User-defined function returning z_warp in [arcsec] at a given radius in [arcsec].
- **return_idx** (*Optional[bool]*) – If True, return the indices of the nearest pixels rather than the interpolated values.

**Returns**

**Either the sky plane x-coordinate in [arcsec] or the** index of the closest pixel.

**y (float/int): Either the sky plane y-coordinate in [arcsec] or the** index of the closest pixel.

**Return type** x (float/int)

**shift_center**(*dx0=0.0*, *dy0=0.0*, *data=None*, *save=True*)

Shift the source center by `dx0` [arcsec] and `dy0` [arcsec] in the x- and y-directions, respectively.

**Parameters**

- **dx0** (*Optional[float]*) – Shfit along the x-axis in [arcsec].
- **dy0** (*Optional[float]*) – Shifta long the y-axis in [arcsec].
- **data** (*Optional[ndarray]*) – Data to shift if not the attached data.
- **save** (*Optional[bool]*) – Whether to overwrite the attached data with the shifted data. If not, return the shifted array. Default is `True`.

**Returns** Shifted array if `save=False`.

**Return type** ndarray

**rotate_image**(*PA*, *data=None*, *save=True*)

Rotate the image such that the red-shifted axis aligns with the x-axis. This is particularly useful for the *emission_height()*.

**Parameters**

- **PA** (*float*) – Position angle of the disk, measured to the red-shifted major axis of the disk, anti-clockwise from North, in [deg].
- **data** (*Optional[ndarray]*) – Data to rotate if not the attached data.
- **save** (*Optional[bool]*) – Whether to overwrite the attached data with the rotated data. If not, return the rotated array. Default is `True`.

**Returns** Rotated array if `save=False`.

**Return type** ndarray

**clip_velocity**(*vmin=None*, *vmax=None*)

Clip the cube between the defined velocity ranges. Will update all necessary values.

**Parameters**

- **vmin** (*Optional[float]*) – Minimum velocity value to include in [m/s].
- **vmax** (*Optional[float]*) – Maximum velocity value to include in [m/s].

**clip_frequency**(*fmin=None*, *fmax=None*)

Clip the cube between the defined frequency ranges. Will update all necessary values.

**Parameters**

- **fmin** (*Optional[float]*) – Minimum frequency to include in [Hz].
- **fmax** (*Optional[float]*) – Maximum frequency to include in [Hz].

**radial_profile**(*rpnts=None*, *rbins=None*, *x0=0.0*, *y0=0.0*, *inc=0.0*, *PA=0.0*, *z0=0.0*, *psi=1.0*, *z1=0.0*, *phi=1.0*, *w_i=0.0*, *w_r=1.0*, *w_t=0.0*, *z_func=None*, *w_func=None*, *PA_min=None*, *PA_max=None*, *exclude_PA=False*, *beam_spacing=False*, *data=None*, *collapse='max'*, *clip_values=None*, *statistic='mean'*, *uncertainty='stddev'*, ***kwargs*)

Returns a radial profile of the data, taking into account the geometry of the disk, including any non-zero emission heights with a description of the parameters found in *disk_coords()*. If the data is 3D, then it is collapsed along the spectral axis with some provided function. More information on the collapsing can be found in *collapse_cube()*.

**Parameters**

- **rpnts** (*ndarray*) – Bin centers in [arcsec].
- **rbins** (*ndarray*) – Bin edges in [arcsec]. NOTE: Only *rpnts* or *rbins* needs to be specified.
- **x0** (*Optional[float]*) – Source right ascension offset [arcsec].
- **y0** (*Optional[float]*) – Source declination offset [arcsec].
- **inc** (*Optional[float]*) – Source inclination [deg].
- **PA** (*Optional[float]*) – Source position angle [deg]. Measured between north and the red-shifted semi-major axis in an easterly direction.
- **z0** (*Optional[float]*) – Aspect ratio at 1" for the emission surface. To get the far side of the disk, make this number negative.
- **psi** (*Optional[float]*) – Flaring angle for the emission surface.

- **z1** (*Optional[float]*) – Aspect ratio correction term at 1" for the emission surface. Should be opposite sign to z0.
- **phi** (*Optional[float]*) – Flaring angle correction term for the emission surface.
- **PA_min** (*Optional[float]*) – Minimum polar angle of the segment of the annulus in [degrees]. Note this is the polar angle, not the position angle.
- **PA_max** (*Optional[float]*) – Maximum polar angle of the segment of the annulus in [degrees]. Note this is the polar angle, not the position angle.
- **exclude_PA** (*Optional[bool]*) – If True, exclude the provided polar angle range rather than include.
- **data** (*Optional[ndarray]*) – Data to use to create the profile, if not the attached data array.
- **collapse** (*Optional[str]*) – Method used to collapse 3D data. Must be 'max' to take the maximum value, 'sum' to sum along the spectral axis or 'int' to integrate along the spectral axis.
- **clip_values** (*Optional[float/iterable]*) – Clip the data values. If a single value is given, clip all values below this, if two values are given, clip values between them.
- **statistic** (*Optional[str]*) – Statistic to use to determin the bin value, either 'mean' or 'median'.
- **uncertainty** (*Optional[str]*) – Measure of the bin uncertainty. Either `'std'` for the standard deviation or `'percentiles'` for the 16th to 84th percentile range about the median. You can also use `'beam'` to divide through by the square root of the number of beams.

**Returns** Bin centers [arcsec], bin statistics and bin uncertainties.

**Return type** [3 x N] ndarray

**collapse_cube**(*method='max'*, *clip_values=None*)

Collapse the 3D cube to a 2D image using the requested method. Three methods are available: `'max'` takes the maximum value along the spectral dimension, `'sum'` sums up the values along the spectral dimension and `'int'` integrates along the spectral dimension.

If you want more flexibility in the way to collapse the cube, we recommend bettermoments.

**Parameters**

- **method** (*Optional[str]*) – Method used to collapse the cube, either `'max'`, `'sum'` or `'int'`.
- **clip_values** (*Optional[tuple/float]*) – Value to clip the data with. If just a single value is provided, clip all values below this, if a `tuple` is provided, clip values between these two values.

**Returns** Collapsed cube as a 2D array.

**Return type** array

**radial_sampling**(*rbins=None*, *rvals=None*, *spacing=0.25*)

Return bins and bin center values. If the desired bin edges are known, will return the bin edges and vice versa. If neither are known will return default binning with the desired spacing.

**Parameters**

- **rbins** (*optional[list]*) – List of bin edges.
- **rvals** (*optional[list]*) – List of bin centers.

- **spacing** (*optional[float]*) – Spacing of bin centers in units of beam major axis.

**Returns** List of bin edges and bin centers.

**Return type** list, list

**pix_per_beam**
Number of pixels per beam.

**beam_per_pix**
Number of beams per pixel.

**beam_area_arcsec**
Beam area in square arcseconds.

**beam_area_str**
Beam area in steradians.

**beam**
Returns the beam parameters in [arcsec], [arcsec], and [degrees].

**convolve_cube**(*bmaj=None*, *bmin=None*, *bpa=None*, *nbeams=1.0*, *fast=True*, *data=None*)
Convolve the cube with a 2D Gaussian beam.

**Parameters**

- **bmaj** (*Optional[float]*) – FWHM of the Gaussian's major axis in [arcsec]. Default is to use the beam major axis.
- **bmin** (*Optional[float]*) – FWHM of the Gaussian's minor axis in [arcsec]. Default is to use the beam minor axis.
- **bpia** (*Optional[float]*) – Position angle of the Gaussian in [degrees]. Default is to use the beam position angle.
- **nbeams** (*Optional[float]*) – Number of beams to convolve the image by. This is simple a multiplicative factor for `bmaj` and `bmin` if they are not provided.
- **fast** (*Optional[bool]*) – Whether to use the FFT method for the convolution. Default is `True`.
- **data** (*Optional[array]*) –

**Returns**

**An array, the same shape as `self.data`, which has been** convolved with the provided beam.

**Return type** ndarray

**add_correlated_noise**(*rms*, *bmaj*, *bmin=None*, *bpa=0.0*, *nchan=2*)
Add the output of correlated_nosie() directly to self.data.

**correlated_noise**(*rms*, *bmaj*, *bmin=None*, *bpa=0.0*, *nchan=2*)
Generate a 3D cube of spatially and spectrall correlated noise, following function from Ryan Loomis. TODO: Allow for a user-defined kernel for the spectral convolution.

**Parameters**

- **rms** (*float*) – Desired RMS of the noise.
- **bmaj** (*float*) – Beam major axis for the spatial convolution in [arcsec].
- **bmin** (*optional[float]*) – Beam minor axis for the spatial convolution in [arcsec]. If no value is provided we assume a circular beam.

- **bpa** (*optional[float]*) – Position angle of the beam, east of north in [degrees]. This is not required for a circular beam.
- **nchan** (*optional[int]*) – Width of Hanning kernel for spectral convolution. By default this is 2.

**Returns**

**An array of noise with the same shape of** the data with a standard deviation provided.

**Return type** noise (ndarray[float])

**extent**
Extent for imshow.

**plotbeam**(*ax*, *x0=0.1*, *y0=0.1*, ***kwargs*)
Plot the sythensized beam on the provided axes.

**Parameters**

- **ax** (*matplotlib axes instance*) – Axes to plot the FWHM.
- **x0** (*float*) – Relative x-location of the marker.
- **y0** (*float*) – Relative y-location of the marker.
- **kwargs** (*dic*) – Additional kwargs for the style of the plotting.

**plotFWHM**(*ax*, *x0=0.125*, *y0=0.125*, *major=True*, *align='center'*, ***kwargs*)
Plot the synthesized beam FWHM on the provided axes.

**Parameters**

- **ax** (*matplotlib axes instance*) – Axes to plot the FWHM.
- **x0** (*float*) – Relative x-location of the marker.
- **y0** (*float*) – Relative y-location of the marker.
- **major** (*bool*) – If True, plot the beam major axis, otherwise the minor axis.
- **align** (*str*) – How to align the marker with respect to the provided x0 value. Must be 'center' (default), 'left' or 'right'.
- **kwargs** (*dic*) – Additional kwargs for the style of the plotting.

**plotaxes**(*ax*, *x0=0.0*, *y0=0.0*, *inc=0.0*, *PA=0.0*, *major=1.0*, ***kwargs*)
Plot the major and minor axes on the provided axis.

**Parameters**

- **ax** (*Matplotlib axes*) – Axes instance to plot onto.
- **x0** (*optional[float]*) – Relative x-location of the center [arcsec].
- **y0** (*optional[float]*) – Relative y-location of the center [arcsec].
- **inc** (*optional[float]*) – Inclination of the disk in [degrees].
- **PA** (*optional[float]*) – Position angle of the disk in [degrees].
- **major** (*optional[float]*) – Size of the major axis line in [arcsec].

**plot_surface**(*ax=None*, *x0=0.0*, *y0=0.0*, *inc=0.0*, *PA=0.0*, *z0=0.0*, *psi=0.0*, *z1=0.0*, *phi=1.0*, *r_min=0.0*, *r_max=None*, *ntheta=9*, *nrad=10*, *check_mask=True*, ***kwargs*)
Overplot the emission surface onto an axis.

**Parameters**

- **ax** (*Optional[AxesSubplot]*) – Axis to plot onto.
- **x0** (*Optional[float]*) – Source right ascension offset [arcsec].
- **y0** (*Optional[float]*) – Source declination offset [arcsec].
- **inc** (*Optional[float]*) – Source inclination [deg].
- **PA** (*Optional[float]*) – Source position angle [deg]. Measured between north and the red-shifted semi-major axis in an easterly direction.
- **z0** (*Optional[float]*) – Aspect ratio at 1" for the emission surface. To get the far side of the disk, make this number negative.
- **psi** (*Optional[float]*) – Flaring angle for the emission surface.
- **z1** (*Optional[float]*) – Aspect ratio correction term at 1" for the emission surface. Should be opposite sign to z0.
- **phi** (*Optional[float]*) – Flaring angle correction term for the emission surface.
- **r_min** (*Optional[float]*) – Inner radius to plot, default is 0.
- **r_max** (*Optional[float]*) – Outer radius to plot.
- **ntheta** (*Optional[int]*) – Number of theta contours to plot.
- **nrad** (*Optional[int]*) – Number of radial contours to plot.
- **check_mask** (*Optional[bool]*) – Mask regions which are like projection errors for highly flared surfaces.

**Returns** Axis with the contours overplotted.

**Return type** ax (AxesSubplot)

**polar_plot**(*rgrid=None*, *tgrid=None*, *x0=0.0*, *y0=0.0*, *inc=0.0*, *PA=0.0*, *z0=0.0*, *psi=0.0*, *z1=0.0*, *phi=1.0*, *w_i=0.0*, *w_r=1.0*, *w_t=0.0*, *z_func=None*, *w_func=None*, *data=None*, *collapse='max'*, *clip_values=None*, *griddata_kwargs=None*, *ax=None*, *xaxis='radius'*, *imshow_kwargs=None*)

Plots the polar deprojection (using *deproject_data_polar()*) of the attached data. You can also specify your own data if you want to.

**Parameters**

- **rgrid** (*Optional[array]*) – Radial
- **x0** (*Optional[float]*) – Source right ascension offset [arcsec].
- **y0** (*Optional[float]*) – Source declination offset [arcsec].
- **inc** (*Optional[float]*) – Source inclination [deg].
- **PA** (*Optional[float]*) – Source position angle [deg]. Measured between north and the red-shifted semi-major axis in an easterly direction.
- **z0** (*Optional[float]*) – Aspect ratio at 1" for the emission surface. To get the far side of the disk, make this number negative.
- **psi** (*Optional[float]*) – Flaring angle for the emission surface.
- **z1** (*Optional[float]*) – Aspect ratio correction term at 1" for the emission surface. Should be opposite sign to `z0`.
- **phi** (*Optional[float]*) – Flaring angle correction term for the emission surface.
- **w_i** (*Optional[float]*) – Warp inclination in [degrees] at the disk center.

- **w_r** (*Optional[float]*) – Scale radius of the warp in [arcsec].
- **w_t** (*Optional[float]*) – Angle of nodes of the warp in [degrees].
- **z_func** (*Optional[callable]*) – User-defined function returning z in [arcsec] at a given radius in [arcsec].
- **w_func** (*Optional[callable]*) – User-defined function returning z_warp in [arcsec] at a given radius in [arcsec].
- **data** (*Optional[array]*) – Data to deproject, otherwise use the attached data. If providing data, it must already be collapsed to a 2D array.
- **collapse** (*Optional[str]*) – Method used to collapse 3D data. Must be `'max'` to take the maximum value, `'sum'` to sum along the spectral axis or `'int'` to integrate along the spectral axis.
- **clip_values** (*Optional[float/iterable]*) – Clip the data values. If a single value is given, clip all values below this, if two values are given, clip values between them.
- **ax** (*Optional[Matplolib axis]*) – Axes to plot the data on.
- **xaxis** (*Optional[str]*) – Which value to have along the x-axis, either `'radius'` or `'polar angle'`.
- **imshow_kwargs** (*Optional[dict]*) – Kwargs to be passed to imshow.

**Returns** Axis on which the plot is plotted.

**Return type** ax (Matoplot axis)

**subtract_continuum**(*continuum=None*, *channels=None*, *N=5*, *data=None*, *save=True*)

Simple continuum subtracted. Model the continuum as the average of either the first and last `N` channels, or, if `channels` is provided, those channels. Then subtract this from each channel.

**Parameters**

- **continuum** (*Optional[ndarray]*) – Model of the continuum to subtract.
- **channels** (*Optional[array]*) – A mask of same size as `velax` defining which channels contain only continuum.
- **N** (*Optional[int]*) – Number of first and final channels to model the continuum with if `continuum` and `channels` not provided.
- **data** (*Optional[ndarray]*) – Data to subtract continuum from if not the attached data.
- **save** (*Optional[bool]*) – If `True`, save the continuum subtracted data as `self.data`, otherwise return it.

**Returns** Continuum subtracted datacube.

**Return type** ndarray

**cross_section**(*x0=0.0*, *y0=0.0*, *PA=0.0*, *mstar=1.0*, *dist=100.0*, *vlsr=None*, *grid=True*, *grid_spacing=None*, *downsample=1*, *cylindrical_rotation=False*, *clip_noise=True*, *min_npnts=5*, *statistic='mean'*, *mask_velocities=None*)

Return the cross section of the data following Dutrey et al. (2017). This yields `I_nu(r, z)`. If `grid=True` then this will be gridded using `scipy.interpolate.griddata` onto axes with the same pixel spacing as the attached data.

**Parameters**

- **x0** (*Optional[float]*) – Source right ascension offset [arcsec].

- **y0** (*Optional[float]*) – Source declination offset [arcsec].
- **PA** (*Optional[float]*) – Position angle of the disk in [deg].
- **mstar** (*Optional[float]*) – Mass of the central star in [Msun].
- **dist** (*Optional[float]*) – Distance to the source in [pc].
- **vlsr** (*Optional[float]*) – Systemic velocity in [m/s]. If `None`, assumes the central velocity.
- **grid** (*Optional[bool]*) – Whether to grid the coordinates to a regular grid. Default is `True`.
- **grid_spacing** (*Optional[float]*) – The spacing, in [arcsec], for the R and Z grids. If `None` is provided, will use pixel spacing.
- **downsample** (*Optional[int]*) – If provided, downsample the coordinates to grid by this factor to speed up the interpolation for large datasets. Default is `1`.
- **cylindrical_rotation** (*Optional[bool]*) – If `True`, assume that the Keplerian rotation decreases with height above the midplane.
- **clip_noise** (*Optional[bool]*) – If `True`, remove all pixels which fall below 3 times the standard deviation of the two edge channels. If the argument is a `float`, use this as the clip level.
- **min_npnts** (*Optional[int]*) – Number of minimum points in each bin for the average. Default is 5.
- **statistic** (*Optional[str]*) – Statistic to calculate for each bin. Note that the uncertainty returned will only make sense with `'max'`, `'mean'` or `'median'`.
- **mask_velocities** (*Optional[list of tuples]*) – List of `(v_min, v_max)` tuples to mask (i.e. remove from the averaging).

**Returns** Either two 1D arrays containing `(r, z, I_nu)`, or, if `grid=True`, two 1D arrays with the `r` and `z` axes and two 2D array of `I_nu` and `dI_nu`.

**Return type** ndarray

**get_cut**(*z=0.0*, *dz=None*, *x0=0.0*, *y0=0.0*, *PA=0.0*, *mstar=1.0*, *dist=100.0*, *vlsr=None*, *grid=True*, *downsample=1*, *griddata_kwargs=None*, *cylindrical_rotation=False*, *clip_noise=True*, *min_npnts=5*, *mask_velocities=None*, *grid_spacing=None*, *statistic='mean'*)

Return a deprojected cut of the data following Matra et al. (2017), which will be `I_nu(x, |y|)`. If `grid=True` then this will be gridded using `scipy.interpolate.griddata` onto axes with the same pixel spacing as the attached data.

**Parameters**

- **z** (*Optional[float]*) – Height at which to take the slice in [arcsec].
- **(Optional[int]** (*dz*) – Width of the slice to take in [arcsec].
- **x0** (*Optional[float]*) – Source right ascension offset [arcsec].
- **y0** (*Optional[float]*) – Source declination offset [arcsec].
- **PA** (*Optional[float]*) – Position angle of the disk in [deg].
- **mstar** (*Optional[float]*) – Mass of the central star in [Msun].
- **dist** (*Optional[float]*) – Distance to the source in [pc].
- **vlsr** (*Optional[float]*) – Systemic velocity in [m/s]. If `None`, assumes the central velocity channel.

- **grid** (*Optional[bool]*) – Whether to grid the coordinates to a regular grid. Default is `True`.
- **downsample** (*Optional[int]*) – If provided, downsample the coordinates to grid by this factor to speed up the interpolation for large datasets. Default is `1`.
- **griddata_kwargs** (*Optional[dic]*) – Dictionary of kwargs to pass to `scipy.interpolate.griddata`.
- **cylindrical_rotation** (*Optional[bool]*) – If `True`, assume cylindrical rotation rather than a Keplerian rotation profile which decreases with height in the disk.
- **clip_noise** (*Optional[bool]*) – If `True`, remove all pixels which fall below 3 times the standard deviation of the two edge channels. If the argument is a `float`, use this as the clip level.

**Returns** Either three 1D arrays containing `(x, y, I_nu)`, or, if `grid=True`, two 1D arrays with the `x` and `|y|` axes and one 2D array of `I_nu`.

**Return type** ndarray

**deproject_data_polar**(*rgrid=None*, *tgrid=None*, *x0=0.0*, *y0=0.0*, *inc=0.0*, *PA=0.0*, *z0=0.0*, *psi=0.0*, *z1=0.0*, *phi=1.0*, *w_i=0.0*, *w_r=1.0*, *w_t=0.0*, *z_func=None*, *w_func=None*, *data=None*, *collapse='max'*, *clip_values=None*, *griddata_kwargs=None*)

Deproject the data into (r, theta) coordinates based on the geometrical properties provided.

**Parameters**

- **x0** (*Optional[float]*) – Source right ascension offset [arcsec].
- **y0** (*Optional[float]*) – Source declination offset [arcsec].
- **inc** (*Optional[float]*) – Source inclination [deg].
- **PA** (*Optional[float]*) – Source position angle [deg]. Measured between north and the red-shifted semi-major axis in an easterly direction.
- **z0** (*Optional[float]*) – Aspect ratio at 1" for the emission surface. To get the far side of the disk, make this number negative.
- **psi** (*Optional[float]*) – Flaring angle for the emission surface.
- **z1** (*Optional[float]*) – Aspect ratio correction term at 1" for the emission surface. Should be opposite sign to z0.
- **phi** (*Optional[float]*) – Flaring angle correction term for the emission surface.
- **w_i** (*Optional[float]*) – Warp inclination in [degrees] at the disk center.
- **w_r** (*Optional[float]*) – Scale radius of the warp in [arcsec].
- **w_t** (*Optional[float]*) – Angle of nodes of the warp in [degrees].
- **z_func** (*Optional[callable]*) – User-defined function returning z in [arcsec] at a given radius in [arcsec].
- **w_func** (*Optional[callable]*) – User-defined function returning z_warp in [arcsec] at a given radius in [arcsec].
- **data** (*Optional[array]*) – Data to deproject, otherwise use the attached data. If providing data, it must already be collapsed to a 2D array.
- **collapse** (*Optional[str]*) – Method used to collapse 3D data. Must be 'max' to take the maximum value, 'sum' to sum along the spectral axis or 'int' to integrate along the spectral axis.

- **clip_values** (*Optional[float/iterable]*) – Clip the data values. If a single value is given, clip all values below this, if two values are given, clip values between them.
- **griddata_kwargs** (*Optional[dict]*) – Kwargs to be passed to gridddata.

**Returns** Radial grid to deproject onto in [arcsec]. tgrid: Polar angle grid to deproject onto in [degrees]. dgrid: Deprojected data array.

**Return type** rgrid

**emission_height**(*inc, PA, x0=0.0, y0=0.0, chans=None, threshold=0.95, smooth=[0.5, 0.5], **kwargs*)

Infer the height of the emission surface following the method in Pinte et al. (2018a).

**Parameters**

- **inc** (*float*) – Inclination of the source in [degrees].
- **PA** (*float*) – Position angle of the source in [degrees].
- **x0** (*Optional[float]*) – Source center offset in x direction in [arcsec].
- **y0** (*Optional[float]*) – Source center offset in y direction in [arcsec].
- **chans** (*Optional[list]*) – The lower and upper channel numbers to include in the fitting.
- **threshold** (*Optional[float]*) – Fraction of the peak intensity at that radius to clip in calculating the data.
- **smooth** (*Optional[list]*) – Kernel to smooth the profile with prior to measuring the peak pixel positions.

**Returns** Deprojected midplane radius in [arcsec]. z (ndarray): Deprojected emission height in [arcsec]. Fnu (ndarray): Intensity of at that location.

**Return type** r (ndarray)

**integrated_spectrum**(*r_min=None, r_max=None, x0=0.0, y0=0.0, inc=0.0, PA=0.0, clip_values=None*)

Return an integrated spectrum in [Jy]. Will convert images in Tb to Jy/beam using the full Planck law. This may cause some noise issues for low SNR data.

**Parameters**

- **r_min** (*Optional[float]*) – Inner radius in [arcsec] of the area to integrate over. Note this is just a circular mask.
- **r_max** (*Optional[float]*) – Outer radius in [arcsec] of the area to integrate over. Note this is just a circular mask.
- **x0** (*Optional[float]*) – Source center offset from the image center in the x direction, measured in [arcsec].
- **y0** (*Optional[float]*) – Source center offset from the image cneter in the y direction, measured in [arcsec].
- **inc** (*Optional[float]*) – Inclination of the circular mask in [degrees], to make it elliptical to integrate over. Default is 0, so results in a circular mask.
- **PA** (*Optional[float]*) – Position angle of the inclined integration mask in [degrees].
- **clip_values** (*Optional[float/iterable]*) – Clip the data values. If a single value is given, clip all values below this, if two values are given, clip values between them.

**Returns** Array of the integrated flux in [Jy] values along the attached velocity axis. `uncertainty` (ndarray): Array of the uncertainties on `flux`.

**Return type** `flux` (ndarray)

**get_deprojected_spectrum**(*r_min*, *r_max*, *PA_min=None*, *PA_max=None*, *exclude_PA=False*, *x0=0.0*, *y0=0.0*, *inc=0.0*, *PA=0.0*, *z0=0.0*, *psi=1.0*, *z1=0.0*, *phi=1.0*, *beam_spacing=False*, *mstar=1.0*, *dist=100.0*, *vrot=None*, *vrad=0.0*, *resample=True*)

Return the azimuthally averaged spectrum from an annulus described by r_min and r_max. The spectra can be deprojected assuming either Keplerian rotation or with the provided vrot and vrad values.

**Parameters** **Coming...** –

**Returns** Spectral axis of the deprojected spectrum. `y` (ndarray[float]): Spectrum, either flux density or brightness temperature depending on the units of the cube. `dy` (ndarray[float]): Uncertainty on each y value based on the scatter in the resampled bins.

**Return type** `x` (ndarray[float])

**keplerian_profile**(*x0=0.0*, *y0=0.0*, *inc=0.0*, *PA=0.0*, *z0=0.0*, *psi=1.0*, *z1=0.0*, *phi=1.0*, *mstar=1.0*, *dist=100.0*, *vlsr=0.0*)

Return a Keplerian rotation profile (for the near side) in [m/s].

**keplerian_curve**(*rpnts*, *mstar*, *dist*, *inc=90.0*, *z0=0.0*, *psi=1.0*, *z1=0.0*, *phi=1.0*)

Return a Keplerian rotation profile [m/s] at rpnts [arcsec].

**Parameters**

- **rpnts** (*ndarray/float*) – Radial locations in [arcsec] to calculate the Keplerian rotation curve at.
- **mstar** (*float*) – Mass of the central star in [Msun].
- **dist** (*float*) – Distance to the source in [pc].
- **inc** (*Optional[float]*) – Inclination of the source in [deg]. If not provided, will return the unprojected value.
- **z0** (*Optional[float]*) – Aspect ratio at 1" for the emission surface. To get the far side of the disk, make this number negative.
- **psi** (*Optional[float]*) – Flaring angle for the emission surface.
- **z1** (*Optional[float]*) – Aspect ratio correction term at 1" for the emission surface. Should be opposite sign to z0.
- **phi** (*Optional[float]*) – Flaring angle correction term for the emission surface.

**Returns**

**Keplerian rotation curve [m/s] at the** specified radial locations.

**Return type** vkep (ndarray/float)

**CLEAN_mask**(*x0=0.0*, *y0=0.0*, *inc=0.0*, *PA=0.0*, *z0=0.0*, *psi=1.0*, *z1=0.0*, *phi=1.0*, *mstar=1.0*, *dist=100.0*, *r_max=None*, *r_min=None*, *vlsr=0.0*, *dV0=500.0*, *dVq=-0.4*, *mask_back=True*, *nbeams=0.0*, *fname=None*, *fast=True*, *return_mask=False*)

Create a mask suitable for CLEANing the data. The flared emission surface is described with the usual geometrical parameters (see disk_coords for more). A radial profile for the line width is also included such that

dV = dV0 * (r / 1")**dVq

providing a little more flexibility to alter the mask shape in the outer regions of the disk.

**Parameters**

- **x0** (*Optional[float]*) – Source right ascension offset [arcsec].
- **y0** (*Optional[float]*) – Source declination offset [arcsec].
- **inc** (*Optional[float]*) – Source inclination [deg].
- **PA** (*Optional[float]*) – Source position angle [deg]. Measured between north and the red-shifted semi-major axis in an easterly direction.
- **z0** (*Optional[float]*) – Aspect ratio at 1" for the emission surface. To get the far side of the disk, make this number negative.
- **psi** (*Optional[float]*) – Flaring angle for the emission surface.
- **z1** (*Optional[float]*) – Aspect ratio correction term at 1" for the emission surface. Should be opposite sign to z0.
- **phi** (*Optional[float]*) – Flaring angle correction term for the emission surface.
- **mstar** (*Optional[float]*) – Mass of the central starr in [Msun].
- **dist** (*Optional[float]*) – Distance to the source in [pc].
- **r_min** (*Optional[float]*) – Inner radius of the disk in [arcsec].
- **r_max** (*Optional[float]*) – Outer radius of the disk in [arcsec].
- **vlsr** (*Optional[float]*) – Systemic velocity in [m/s].
- **dV0** (*Optional[float]*) – Doppler line width at 1 arcsec in [m/s].
- **dVq** (*Optional[float]*) – Power-law exponent of the line width profile. A value of 0 will result in dV which is constant.
- **nbeams** (*Optional[float]*) – The number of beams kernels to convolve the mask with. For example, nbeams=1 will conolve the mask with the attached beam, nbeams=2 will convolve it with a beam double the size.
- **fname** (*Optional[str]*) – File name to save the mask to. If none is specified, will use the same path but with a ''.mask.fits' extension.
- **fast** (*Optional[bool]*) – If True, use the fast convolve from Astropy.
- **return_mask** (*Optional[bool]*) – If True, return the mask as an array, otherwise, save it to a FITS file.
- **Returns** – mask (ndarray): If return_mask is True, will return a mask matching the shape of the attached cube.

**synthetic_obs**(*bmaj=None*, *bmin=None*, *bpa=0.0*, *rms=None*, *chan=None*, *nchan=None*, *rescale='auto'*, *spectral_response=None*, *save=False*)

Generate synthetic observations by convolving the data spatially and spectrally and adding correlated noise. Will automatically convert the attached brightness unit to 'Jy/beam'.

**Parameters**

- **bmaj** (*optional[float]*) – Beam major axis in [arcsec].
- **bmin** (*optional[float]*) – Beam minor axis in [arcsec].
- **bpa** (*optional[float]*) – Beam position angle in [degrees].
- **rms** (*optional[float]*) – RMS noise of the noise.
- **chan** (*optional[float]*) – Channel size (m/s) of the resulting data.

- **nchan** (*optional[int]*) – Number of channels of the resulting data. If this would extend beyond the attached velocity then these edge channels are ignored.
- **rescale** (*optional[int]*) – Rescaling factor for the pixels. If `rescale='auto'` then the pixels will be rescaled so there's 5 pixels per bmin.
- **spectral_response** (*optional[str]*) – Type of spectral response to include. `'hanning'` will include a triangle kernel, while `'averageX'`, where `'X'` is a number will use a simple running average of `X` channels.
- **save** (*optional[bool/str]*) – If True, save the data as a new cube. You may also provide a path to save to (noting that this will overwrite anything).

**Returns**

**If not saved to disk, returns the spatial axis, velocity** axis and the datacube.

**Return type** ndarrays

**get_mask**(*r_min=None*, *r_max=None*, *exclude_r=False*, *PA_min=None*, *PA_max=None*, *exclude_PA=False*, *x0=0.0*, *y0=0.0*, *inc=0.0*, *PA=0.0*, *z0=0.0*, *psi=1.0*, *z1=0.0*, *phi=1.0*, *w_i=0.0*, *w_r=1.0*, *w_t=0.0*, *z_func=None*, *w_func=None*)

Returns a 2D mask for pixels in the given region.

**Parameters**

- **r_min** (*float*) – Minimum midplane radius of the annulus in [arcsec].
- **r_max** (*float*) – Maximum midplane radius of the annulus in [arcsec].
- **exclude_r** (*Optional[float]*) – If True, exclude the provided radial rangle rather than including it.
- **PA_min** (*Optional[float]*) – Minimum polar angle of the segment of the annulus in [degrees]. Note this is the polar angle, not the position angle.
- **PA_max** (*Optional[float]*) – Maximum polar angle of the segment of the annulus in [degrees]. Note this is the polar angle, not the position angle.
- **exclude_PA** (*Optional[bool]*) – If True, exclude the provided polar angle range rather than including it.
- **x0** (*Optional[float]*) – Source right ascension offset (arcsec).
- **y0** (*Optional[float]*) – Source declination offset (arcsec).
- **inc** (*Optional[float]*) – Source inclination (degrees).
- **PA** (*Optional[float]*) – Source position angle (degrees). Measured between north and the red-shifted semi-major axis in an easterly direction.
- **z0** (*Optional[float]*) – Aspect ratio at 1" for the emission surface. To get the far side of the disk, make this number negative.
- **psi** (*Optional[float]*) – Flaring angle for the emission surface.
- **z1** (*Optional[float]*) – Aspect ratio correction term at 1" for the emission surface. Should be opposite sign to z0.
- **phi** (*Optional[float]*) – Flaring angle correction term for the emission surface.
- **z_func** (*Optional[callable]*) – A function which returns the emission height in [arcsec] for a midplane radius in [arcsec]. If provided, will be used in place of the parametric emission surface.

**Returns** A 2D mask.

**Return type** mask (ndarray)

**velocity_resolution**(*dnu*)
Convert spectral resolution in [Hz] to [m/s].

**spectral_resolution**(*dV*)
Convert velocity resolution [m/s] to [Hz].

**velocity_to_restframe_frequency**(*velax=None*, *vlsr=0.0*)
Return the rest-frame frequency [Hz] of the given velocity [m/s].

**restframe_frequency_to_velocity**(*nu*, *vlsr=0.0*)
Return the velocity [m/s] of the given rest-frame frequency [Hz].

**spiral_coords**(*r_p*, *t_p*, *m=None*, *r_min=None*, *r_max=None*, *mstar=1.0*, *T0=20.0*, *Tq=-0.5*, *dist=100.0*, *clockwise=True*, *frame_out='cartesian'*)
Spiral coordinates from Bae & Zhaohuan (2018a). In order to recover the linear spirals from Rafikov (2002), use m >> 1.

**Parameters**

- **r_p** (*float*) – Orbital radius of the planet in [arcsec].
- **t_p** (*float*) – Polar angle of planet relative to the red-shifted major axis of the disk in [radians].
- **m** (*optional[int]*) – Azimuthal wavenumber of the spiral. If not specified, will assume the dominant term based on the rotation and temperature profiles.
- **r_min** (*optional[float]*) – Inner radius of the spiral in [arcsec].
- **r_max** (*optional[float]*) – Outer radius of the spiral in [arcsec].
- **mstar** (*optioanl[float]*) – Stellar mass of the central star in [Msun] to calculate the rotation profile.
- **T0** (*optional[float]*) – Gas temperature in [K] at 1 arcsec.
- **Tq** (*optional[float]*) – Exoponent of the radial gas temperature profile.
- **dist** (*optional[float]*) – Source distance in [pc] used to scale [arcsec] to [au] in the calculation of the rotation profile.
- **clockwise** (*optional[bool]*) – Direction of the spiral.
- **frame_out** (*optional[str]*) – Coordinate frame of the returned values, either 'cartesian' or 'cylindrical'.

**Returns** Coordinates of the spiral in either cartestian or cylindrical frame.

**Return type** ndarray

**plot_axes**(*ax*, *x0=0.0*, *y0=0.0*, *inc=0.0*, *PA=0.0*, *major=1.0*, *plot_kwargs=None*)
Plot the major and minor axes on the provided axis.

**Parameters**

- **ax** (*Matplotlib axes*) – Axes instance to plot onto.
- **x0** (*Optional[float]*) – Relative x-location of the center [arcsec].
- **y0** (*Optional[float]*) – Relative y-location of the center [arcsec].
- **inc** (*Optional[float]*) – Inclination of the disk in [degrees].
- **PA** (*Optional[float]*) – Position angle of the disk in [degrees].
- **major** (*Optional[float]*) – Size of the major axis line in [arcsec].

- **plot_kwargs** (*Optional[dict]*) – Dictionary of parameters to pass to `matplotlib.plot`.

**Returns** Matplotlib ax with axes drawn.

**Return type** matplotlib axis

# Index


## A

add_correlated_noise() (*imgcube.imagecube method*), 11

## B

beam (*imgcube.imagecube attribute*), 11
beam_area_arcsec (*imgcube.imagecube attribute*), 11
beam_area_str (*imgcube.imagecube attribute*), 11
beam_per_pix (*imgcube.imagecube attribute*), 11

## C

CLEAN_mask() (*imgcube.imagecube method*), 18
clip_frequency() (*imgcube.imagecube method*), 9
clip_velocity() (*imgcube.imagecube method*), 9
collapse_cube() (*imgcube.imagecube method*), 10
convolve_cube() (*imgcube.imagecube method*), 11
correlated_noise() (*imgcube.imagecube method*), 11
cross_section() (*imgcube.imagecube method*), 14

## D

deproject_data_polar() (*imgcube.imagecube method*), 16
disk_coords() (*imgcube.imagecube method*), 4
disk_to_sky() (*imgcube.imagecube method*), 7

## E

emission_height() (*imgcube.imagecube method*), 17
extent (*imgcube.imagecube attribute*), 12

## G

get_annulus() (*imgcube.imagecube method*), 5
get_cut() (*imgcube.imagecube method*), 15
get_deprojected_spectrum() (*imgcube.imagecube method*), 18
get_mask() (*imgcube.imagecube method*), 20
get_vlos() (*imgcube.imagecube method*), 6

## I

imagecube (*class in imgcube*), 3
integrated_spectrum() (*imgcube.imagecube method*), 17

## K

keplerian_curve() (*imgcube.imagecube method*), 18
keplerian_profile() (*imgcube.imagecube method*), 18

## P

pix_per_beam (*imgcube.imagecube attribute*), 11
plot_axes() (*imgcube.imagecube method*), 21
plot_surface() (*imgcube.imagecube method*), 12
plotaxes() (*imgcube.imagecube method*), 12
plotbeam() (*imgcube.imagecube method*), 12
plotFWHM() (*imgcube.imagecube method*), 12
polar_plot() (*imgcube.imagecube method*), 13

## R

radial_profile() (*imgcube.imagecube method*), 9
radial_sampling() (*imgcube.imagecube method*), 10
restframe_frequency_to_velocity() (*imgcube.imagecube method*), 21
rotate_image() (*imgcube.imagecube method*), 8

## S

shift_center() (*imgcube.imagecube method*), 8
sky_to_disk() (*imgcube.imagecube method*), 6
spectral_resolution() (*imgcube.imagecube method*), 21
spiral_coords() (*imgcube.imagecube method*), 21
subtract_continuum() (*imgcube.imagecube method*), 14
synthetic_obs() (*imgcube.imagecube method*), 19

## V

velocity_resolution() (*imgcube.imagecube method*), 21
velocity_to_restframe_frequency() (*imgcube.imagecube method*), 21