kinmspytest

Release 2.0.10

Apr 08, 2020

Contents:

Note: Current version: 2.0.10

Welcome to the KinMSpy documentation. Here you will find information regarding the ins-and-outs of operating KinMSpy.

If you have found this software useful for your research, We would appreciate an acknowledgment to the use of the "KINematic Molecular Simulation (KinMS) routines of Davis et al., (2013)". MNRAS, Volume 429, Issue 1, p.534-555.

This software is provided as is without any warranty whatsoever. For details of permissions granted please see the official license.

---

class KinMS.KinMS_core.**KinMSError**

Generates errors under the flag 'KinMSError'.

Class KinMSError Instantiates the Exception error 'KinMSError', for warning the user of faults and exceptions.

class KinMS.KinMS_core.**KinMS**(*xs*, *ys*, *vs*, *cellSize*, *dv*, *beamSize*, *inc*, *posAng*, *gasSigma=0*, *diskThick=0*, *flux_clouds=0*, *sbProf=[]*, *sbRad=[]*, *velRad=[]*, *velProf=[]*, *inClouds=[]*, *vLOS_clouds=[]*, *massDist=[]*, *vRadial=[]*, *ra=None*, *dec=None*, *nSamps=None*, *seed=None*, *intFlux=None*, *vSys=None*, *phaseCent=None*, *vOffset=None*, *vPosAng=None*, *vPhaseCent=None*, *restFreq=None*, *fileName=''*, *fixSeed=False*, *cleanOut=False*, *returnClouds=False*, *huge_beam=False*, *verbose=False*, *toplot=False*)

Class KinMS Main workhorse of KinMSpy, used to gerenate spectral cubes.

Parameters

- **xs** – (float or int) x-axis size for resultant cube (in arcseconds)
- **ys** – (float or int) y-axis size for resultant cube (in arcseconds)
- **vs** – (float or int) Velocity axis size for resultant cube (in km/s)
- **cellSize** – (float or int) Pixel size required (arcsec/pixel)
- **dv** – (float or int) Channel size in velocity direction (km/s/channel)
- **beamSize** – (float or int, or list or array of float or int) Scalar or three element list for size of convolving beam (in arcseconds). If a scalar then beam is assumed to be circular. If a list/array of length two. these are the sizes of the major and minor axes, and the position angle is assumed to be 0. If a list/array of length 3, the first 2 elements are the major and minor beam sizes, and the last the position angle (i.e. [bmaj, bmin, bpa]).
- **inc** – (float or int, or list or array of float or int) Inclination angle of the gas disc on the sky (degrees). Can input a constant or a vector, giving the inclination as a function of the radius vector 'velrad' (in order to model warps etc).
- **posAng** – (float or int, or list or array of float or int) Position angle (PA) of the disc (a PA of zero means that the redshifted part of the cube is aligned with the positive y-axis). If single valued then the disc major axis is straight. If an array is passed then it should describe how the position angle changes as a function of *velrad* (so this can be used to create position angle warps).
- **gasSigma** – (float or int, or array or list of float or int) Optional, default is value 0. Velocity dispersion of the gas (units of km/s). If single valued then the velocity dispersion

is constant throughout the disc. If an array/list is passed then it should describe how the velocity dispersion changes as a function of 'velrad'.

- **diskThick** – (float or int, or array or list of float or int) Optional, default value is 0. The disc scaleheight in arcseconds. If a single value then this is used at all radii. If an array/list then it should have the same length as 'sbrad', and will be the disc thickness as a function of that.
- **flux_clouds** – (array or list of float or int) Optional, default value is 0. This vector can be used to supply the flux of each point in 'inclouds'. If used alone then total flux in the model is equal to total(flux_inclouds). If 'intflux' used then this vector denotes the relative brightness of the points in 'inclouds'.
- **sbProf** – (array or list of float or int) Optional, default value is []. Surface brightness profile (arbitrarily scaled) as a function of 'sbrad'.
- **sbRad** – (array or list of float or int) Optional, default value is []. Radius vector for surface brightness profile (units of arcseconds).
- **velRad** – (array or list of float or int) Optional, defaults to 'sbRad'. Radius vector for velocity profile (units of arcseconds).
- **velProf** – (array or list of float or int) Optional, default value is []. Circular velocity profile (in km/s) as a function of 'velrad'.
- **inClouds** – (array or list of float or int) Optional, default value is []. If your required gas distribution is not symmetric, you may input vectors containing the position of the clouds you wish to simulate. This 3-vector should contain the x, y and z positions, in units of arcseconds from the phase centre. If this variable is used, then 'diskthick', 'sbrad' and 'sbprof' are ignored. Example: inclouds = [[0,0,0], [10,-10,2], . . . , [xpos, ypos, zpos]].
- **vLOS_clouds** – (array or list of float or int) Optional, default value is []. This vector should contain the LOS velocity for each point defined in 'inclouds', in units of km/s. If not supplied then 'inclouds' is assumed to be the -face on- distribution and that 'velprof' or 'velrad' should be used, and the distribution projected. If this variable is used then 'gassigma' and 'inc' are ignored.
- **massDist** – (list of float or int) Optional, default value is []. List of [gasmass, distance] - total gas mass in solar masses, total distance in Mpc.
- **vRadial** – (float or int, or array or list of float or int) Optional, default value is 0. Magnitude of inflow/outflowing motions (km/s). Negative numbers are inflow, positive numbers denote outflow. These are included in the velocity field using formalism of 'kinemetry' (Krajnović et al. 2006 MNRAS, 366, 787). Can input a constant or a vector, giving the radial motion as a function of the radius vector 'velrad'.
- **ra** – (float) Optional, default value is None. RA to use in the header of the output cube (in degrees).
- **dec** – (float) Optional, default value is None. Dec to use in the header of the output cube (in degrees).
- **nSamps** – (float or int) Optional, default value is 1e5. Number of cloudlets to use to create the model. Large numbers will reduce numerical noise (especially in large cubes), at the cost of increasing runtime.
- **seed** – (array or list of float or int) Optional, default value is [100, 101, 102, 103]. List of length 4 containing the seeds for random number generation.
- **intFlux** – (float) Optional, default value is 0. Total integrated flux you want the output gas to have. (In Jy/km/s).

- **vSys** – (float) Optional, default value is None. Systemic velocity (km/s).
- **phaseCent** – (list or array of float or int of length 2) Optional, default value is [0, 0]. Specifies the morphological centre of the disc structure you create with respect to the central pixel of the generated cube.
- **vOffset** – (float or int) Optional, default value is 0. Offset from the centre of the velocity axis in km/s.
- **vPosAng** – (float or int, or array or list of float or int) Optional, default value is 0. Kinematic position angle of the disc, using the usual astronomical convention. If single valued then the disc kinematic major axis is straight. If an array is passed then it should describe how the kinematic position angle changes as a function of 'velrad'. Used if the kinematic and morphological position angles are not the same.
- **vPhaseCent** – (list of float or int of length 2) Optional, default value is [0, 0]. Kinematic centre of the rotation in the x-y plane. Units of pixels. Used if the kinematic and morphological centres are not the same.
- **restFreq** – (float) Optional, default value = 115.271e9 (12CO(1-0)). Rest frequency of spectral line of choice (in Hz). Only matters if you are outputting a FITS file.
- **fileName** – (str) Optional, default value is ''. If you wish to save the resulting model to a fits file, set this variable. The output filename will be 'filename'_simcube.fits
- **fixSeed** – (bool) Whether to use a fixed (or random) seed (list of four integers).
- **cleanOut** – (bool) Optional, default value is False. If True then do not convolve with the beam, and output the "clean components". Useful to create input for other simulation tools (e.g sim_observe in CASA).
- **returnClouds** – (bool) Optional, default value is False. If set True then KinMS returns the created 'inclouds' and 'vlos_clouds' in addition to the cube.
- **huge_beam** – (bool) Optional, default is False. If True then astropy's convolve_fft is used instead of convolve, which is faster for very large beams.
- **pool** – (bool) Optional, default is False. If True then the convolution is performed parallelly to speed up the code.
- **verbose** – (bool) Optional, default is False. If True, messages are printed throughout the code.
- **toplot** – (bool) Optional, default if False. If True, moment 0 and 1 maps, and a PVD and spectrum of the output cube are plotted.

add_fluxes(*clouds2do*, *subs*, *x_size*, *y_size*, *v_size*)

If there are clouds to use, and we know the flux of each cloud, add them to the cube. If not, bin each position to get a relative flux.

Parameters

- **clouds2do** – (ndarray) contains the x-, y-, and v-positions of the cloudslets in the cube
- **subs** – (ndarray) the indices of the cloudlets in the cube
- **x_size** – (int) size of the cube in the x-direction
- **y_size** – (int) size of the cube in the y-direction
- **v_size** – (int) size of the cube in the v-direction

Returns spectral cube with fluxes added to the cloudlets

create_warp(*array*, *r_flat*)
If the array provided has a length > 1, create a warp. If it's a single value, create a flat profile.

Parameters

- **array** – (ndarray) array containing the radial profile
- **r_flat** – (ndarray) Radius of each cloudlet from the kinematic centre in the plane of the disc (units of pixels)

Returns ndarray with the radial profile of the disc

find_clouds_in_cube(*los_vel*, *cent*, *x2*, *y2*, *x_size*, *y_size*, *v_size*)
Returns the clouds that lie inside the cube.

Parameters

- **los_vel** – (ndarray) contains the line of sight velocities of each cloudlet, in km/s.
- **cent** – (ndarray of length 2) contains the x and y coordinates of the centre of the object within the cube
- **x2** – (ndarray) x-positions of the cloudlets within the cube
- **y2** – (ndarray) y-positions of the cloudlets within the cube
- **x_size** – (int) size of the cube in the x-direction
- **y_size** – (int) size of the cube in the y-direction
- **v_size** – (int) size of the cube in the z-direction

Returns arrays with the positions of the cloudlets within the cube, and the indices of these positions

gasGravity_velocity(*x_pos*, *y_pos*, *z_pos*, *massDist*, *velRad*)
Calculates an array of line-of-sight velocity alterations, accounting for the effects of internal gas in the disk.

Parameters

- **x_pos** – (numpy array) X position of each cloudlet. Units of arcseconds.
- **y_pos** – (numpy array) Y position of each cloudlet. Units of arcseconds.
- **z_pos** – (numpy array) Z position of each cloudlet. Units of arcseconds.
- **massDist** – (numpy array) Array of ([gasmass,distance]) - total gas mass in solar masses, total distance in Mpc.
- **velRad** – (numpy array) Radius vector for cloudlets (in units of pixels).

Return add_to_circ_vel (numpy array) Additions to the circular velocity due to the internal mass of the gas, in units of km/s.

generate_cloudlets()
A helper function for generating cloudlets by running kinms_sampleFromArbDist_oneSided. Raises a KinMSError if generate_cloudlets is called but sbRad and sbProf are not.

Returns None

inclination_projection(*ang*, *x1*, *y1*, *z1*)
Apply the projection as a result of inclination to the cloudlets.

Parameters

- **ang** – (float) inclination angle (in degrees)

- **x1** – (ndarray) x-positions of the cloudlets
- **y1** – (ndarray) y-positions of the cloudlets
- **z1** – (ndarray) z-positions of the cloudlets

Returns x-, y-, and z-positions of the projected cloudlets

kinms_create_velField_oneSided(*velRad*, *posAng_rad=None*, *inc_rad=None*)

Creates an array of line-of-sight velocities, accounting for velocity dispersion and projection.

Parameters

- **velRad** – (numpy array) Radius vector for velocity profile (units of arcseconds).
- **posAng_rad** – (float or int, or array of float or int) Optional, default value is None. Position angle (PA) of the disc (a PA of zero means that the redshifted part of the cube is aligned with the positive y-axis). If single valued then the disc major axis is straight. If an array is passed then it should describe how the position angle changes as a function of *velrad* (so this can be used to create position angle warps).
- **inc_rad** – (float or int, or array of float or int) Optional, default value is None. Inclination angle of the gas disc on the sky (degrees). Can input a constant or a vector, giving the inclination as a function of the radius vector 'velrad' (in order to model warps etc).

Return los_vel (numpy array) Line-of-sight velocities for projected particles positioned by velRad.

kinms_sampleFromArbDist_oneSided(*sbRad*, *sbProf*, *nSamps*, *diskThick*, *fixSeed=None*)

Samples cloudlets from radial profiles provided given that inClouds is not provided in the __init__.

Parameters

- **sbRad** – (numpy array) Radius vector for surface brightness profile (units of arcseconds).
- **sbProf** – (numpy array) Surface brightness profile (arbitrarily scaled) as a function of 'sbrad'.
- **nSamps** – (int) Number of cloudlets to use to create the model. Large numbers will reduce numerical noise (especially in large cubes), at the cost of increasing runtime.
- **diskThick** – (numpy array) The disc scaleheight in arcseconds. If a single value then this is used at all radii. If an array/list then it should have the same length as 'sbrad', and will be the disc thickness as a function of that.
- **fixSeed** – (bool) Whether to use a fixed (or random) seed (list of four integers).

Return inClouds (numpy array) 3 dimensional array of cloudlet positions within the cube initialised by KinMS().

makebeam()

Creates a psf with which one can convolve their cube based on the beam provided.

Return psf or trimmed_psf (float array) psf required for convlution in self.model_cube(). trimmed_psf returned if self.huge_beam=False, otherwise default return is the untrimmed psf.

model_cube()

Do the actual modelling of the spectral cube

Returns ~~the cube~~

normalise_cube(*cube*, *psf*)

Normalise cube by the known integrated flux.

Parameters

- **cube** – (3D array) unnormalised spectral cube
- **psf** – (2D array) psf of the mock observations, to convolve the cube with

position_angle_rotation(*ang*, *x2*, *y2*, *z2*)
Apply the projection as a result of the position angle to the cloudlets.

Parameters

- **ang** – (float) position angle (in degrees)
- **x2** – (ndarray) x-positions of the cloudlets
- **y2** – (ndarray) y-positions of the cloudlets
- **z2** – (ndarray) z-positions of the cloudlets

Returns x-, y-, and z-positions of the projected cloudlets

print_variables()
If "verbose", prints a summary of parameters for the user's convenience.

Returns (string) formatted display of all parameters used in KinMS() initialisation

save_fits(*cube*, *cent*)
Outputs a .fits file containing the datacube and relevant header information.

Parameters

- **cube** – (numpy array) 3 dimensional spectral cube required for saving to .fits file
- **cent** – (numpy array of intigers) Location of the central x and y positions (in units of pixels), and index of the central velocity channel.

Returns None

set_cloud_positions()
Calculate and return the positions and velocities of the cloudlets in inClouds, and the radial distance in the x and y plane.

Returns None

set_cloud_velocities()
Find the los velocity and cube position of the clouds. If los velocity specified, assume that the clouds have already been projected correctly.

Returns arrays with the x-, y-, and z- positions of the cloudlets, and their los velocities

class KinMS.utils.KinMS_figures.**KinMS_plotter**(*f*, *xsize*, *ysize*, *vsize*, *cellsize*, *dv*, *beamsize*, *posang=None*, *phasecent=None*, *pvdthick=None*, *savepath=None*, *savename=None*, *pdf=True*, *overcube=False*, *title=False*)

Class KinMS_plotter Generates moment maps and position velocity diagrams for input spectral cubes

Parameters

- **f** – (numpy array) Spectral cube used to generate plots
- **xsize** – (float or int) x-axis size for resultant cube (in arcseconds)
- **ysize** – (float or int) y-axis size for resultant cube (in arcseconds)
- **vsize** – (float or int) Velocity axis size for resultant cube (in km/s)

- **cellSize** – (float or int) Pixel size required (arcsec/pixel)
- **dv** – (float or int) Channel size in velocity direction (km/s/channel)
- **beamSize** – (float or int, or list or array of float or int) Scalar or three element list for size of convolving beam (in arcseconds). If a scalar then beam is assumed to be circular. If a list/array of length two. these are the sizes of the major and minor axes, and the position angle is assumed to be 0. If a list/array of length 3, the first 2 elements are the major and minor beam sizes, and the last the position angle (i.e. [bmaj, bmin, bpa]).
- **posang** – (float or int) Position angle (PA) of the disc (a PA of zero means that the red-shifted part of the cube is aligned with the positive y-axis). If single valued then the disc major axis is straight. If an array is passed then it should describe how the position angle changes as a function of *velrad* (so this can be used to create position angle warps).
- **phasecent** – (list or ndarray of length 2) offset between the morphological centre of the emission and the centre of the image. Positive [x, y] means that the morphological centre is to the right and top of the image centre.
- **pvdthick** – UNDER CONSTRUCTION
- **savepath** – (string) path to directory in which the plots are saved to
- **savename** – (string) a filename assigned to the plots when saved
- **pdf** – (bool) Optional, default value is True. saves the plots as a .pdf file
- **overcube** – (bool) UNDER CONSTRUCTION
- **title** – (bool) Optional, default value is False. assigns a title to the plots

makebeam()

Creates a psf with which one can convolve their cube based on the beam provided.

Return psf or trimmed_psf (float array) psf required for convlution in self.model_cube(). trimmed_psf returned if self.huge_beam=False, otherwise default return is the untrimmed psf.

CHAPTER 1

Indices and tables

- genindex
- modindex
- search

Index