pydisc - First Tutorial

March 2, 2020

1 pydisc Notebook - 1.0

1.1 1 - Introduction

pydisc is a python package meant to ease the use the manipulation of maps and profiles and the computation of basic quantities pertaining to galactic Discs. In this Notebook, I will show you how to use the main functionalities of *pydisc*.

1.2 2 - Structures in *pydisc*

1.2.1 DataMaps, DataProfiles, Maps and Profiles

In the language of *pydisc*: - **DataMaps** are data on a grid (hence 2D), e.g., values like velocities, flux, etc. The grid on which it is defined should be regular. - **DataProfiles** are data on a radial profile (hence 1D). - **Maps** are then a **set of** *DataMaps* associated with a set of coordinates X, Y. - **Profiles** are then a **set of** *DataProfiles* associated with a set of radial coordinates R.

DataMaps have orientations defined as 'NE_direct' indicating if the North-to-East axis is direct (counter-clockwise) or indirect (clockwise). It also has an 'alpha_North' angle which provides the angle between the North and the top (positive y-axis). DataMaps also have a pixelscale which provides the conversion between arcseconds and pixels in case the X,Y grids are not defined. If this is the case, X, and Y will be computed using just the indices from the grid.

DataMaps and DataProfiles have 'units' as defined by astropy units. These should be compatible with e.g., arcseconds, so these are observational.

DataMap arguments: - dunit: astropy unit - order: velocity moment order. Hence velocities are order=1, flux or mass is 0, dispersion is 2, anything else would be -1 and there is a category for "dummy" maps with order=-10. - dname: name of the datamap - flag: a flag which is meant to add info (string) - data and edata: numpy arrays. If edata is not provided, it will be defined as None.

DataProfiles have similar arguments, but with punit (profile unit) and pname.

Maps arguments: - name: name of the map - X and Y: the 2 main arrays. If not provided, indices will be used. - Xcen and Ycen: centre for the 0,0 - XYunit: unit (astropy) for the X and Y axis - NE_direct, alpha_North, etc.

Note that a Map can have many datamaps: hence a set of X,Y can have many data associated to it (sharing the same coordinates), each one having a different dname, order, flag etc.

1.2.2 Galaxy

A 'Galaxy' is an object which has some characteristics like: a distance, a Position Angle for the line of Nodes, an inclination (in degrees) and the Position Angle for a bar if present.

1.2.3 GalacticDisc

A 'GalacticDisc' is a structure associating a set of Maps and Profiles and a given Galaxy.

This is the main structure which we will be using for the calculation of various quantities.

There are a number of associated classes, namely: - 'DensityWave': associated with methods for density waves like the Tremaine Weinberg method - 'GalacticTorque': associated with methods for deriving torques

all inheriting from the GalacticDisc class, thus sharing a number of functionalities, but also have their own specific ones (which require a set of maps).

The 'grammar' for maps and datamaps is simple (a priori): - if you have an attribute like "data" you can input this in the argument list as: "data*". Hence if the map is name"MUSE" and the datamap named "vstar" you should have an argument for the data as "dataMUSE_vstar" and the associated "edataMUSE_vstar" if you have uncertainties for this map etc. Same applies for all argument of the maps and data, for example (using the same example): orderMUSE_vstar, XMUSE, YMUSE, XcenMUSE, YcenMUSE, flagMUSE_vstar… - In this way you can have several datamaps attached to a single map and have e.g.,: XMUSE, YMUSE, dataMUSE_vstar, dataMUSE_gas, dataMUSE…*

2 3- Examples

2.1 3.1 - Tremaine Weinberg

Here is an example of how to get a Tremaine-Weinberg calculation made on a set of maps using the *DensityWave* class.

[13]:
```
# Importing the package and the DensityWave class
import pydisc
from pydisc.density_wave import DensityWave

# Importing useful modules
from astropy.io import fits as pyfits
from os.path import join as joinpath
import numpy as np

# Getting the data
ddata = "/home/science/PHANGS/MUSE/MUSEDAP/"
n1512 = "NGC1512_MAPS.fits"
# Open the Maps files
maps = pyfits.open(joinpath(ddata, n1512))
```

```python
# Extract the mass, flux, and velocity maps
mass = maps['STELLAR_MASS_DENSITY'].data
flux = maps['FLUX'].data
vel = maps['V_STARS'].data
```

[14]:
```python
# mname is for mapname.
mydisc = DensityWave(data_flux=flux, edata_flux=np.zeros_like(flux),
                     data_mass=mass, data_vel=vel, edata_vel=np.zeros_like(vel),
                     mname="MUSE", Xcen=462.5, Ycen=464.4, PAnodes=90)
```

```
WARNING: X or Y not provided. Using Pixel XY grid.
INFO: attaching map MUSE
```

[15]:
```python
# We can now look at the structure itself. 'mydisc' has a one map, which is␣
↪named 'MUSE'.
# This map is in a dictionary and is a Map class, as shown when printing it.
mydisc.maps
```

[15]:
```
{'MUSE': <pydisc.disc_data.Map at 0x7f1ab06bedd0>}
```

[16]:
```python
# We can also find out about the other variables:
mydisc.maps['MUSE'].X
```

[16]:
```
array([[-462.5, -461.5, -460.5, …,  440.5,  441.5,  442.5],
       [-462.5, -461.5, -460.5, …,  440.5,  441.5,  442.5],
       [-462.5, -461.5, -460.5, …,  440.5,  441.5,  442.5],
       …,
       [-462.5, -461.5, -460.5, …,  440.5,  441.5,  442.5],
       [-462.5, -461.5, -460.5, …,  440.5,  441.5,  442.5],
       [-462.5, -461.5, -460.5, …,  440.5,  441.5,  442.5]],
      dtype=float32)
```

[17]:
```python
# This Map actually has datamaps as shown here, each one having data.
# You can see that this Map has actually three datamaps, one with flux, one␣
↪with mass, the last one with vel.
mydisc.maps['MUSE'].dmaps
```

[17]:
```
{'flux': <pydisc.disc_data.DataMap at 0x7f1ae3f79150>,
 'mass': <pydisc.disc_data.DataMap at 0x7f1ab06b1d10>,
 'vel': <pydisc.disc_data.DataMap at 0x7f1ae40463d0>}
```

[18]:
```python
# We can call the data like this (note that the array shows the nan from the␣
↪outer part of the map)
mydisc.maps['MUSE'].dmaps['flux'].data
```

[18]:
```
array([[nan, nan, nan, …, nan, nan, nan],
       [nan, nan, nan, …, nan, nan, nan],
```

```
        [nan, nan, nan, …, nan, nan, nan],
        …,
        [nan, nan, nan, …, nan, nan, nan],
        [nan, nan, nan, …, nan, nan, nan],
        [nan, nan, nan, …, nan, nan, nan]])
```

[19]:
```
# or like this using the combined "data" with the name of the data map.
mydisc.maps['MUSE'].dmaps.flux.data
```

[19]:
```
array([[nan, nan, nan, …, nan, nan, nan],
       [nan, nan, nan, …, nan, nan, nan],
       [nan, nan, nan, …, nan, nan, nan],
       …,
       [nan, nan, nan, …, nan, nan, nan],
       [nan, nan, nan, …, nan, nan, nan],
       [nan, nan, nan, …, nan, nan, nan]])
```

[20]:
```
# to make it simpler, the maps and dmaps are merged into one attribute␣
 ↪automatically
mydisc.MUSE_flux
```

[20]:
```
<pydisc.disc_data.DataMap at 0x7f1ae3f79150>
```

[21]:
```
mydisc.MUSE_flux.data
```

[21]:
```
array([[nan, nan, nan, …, nan, nan, nan],
       [nan, nan, nan, …, nan, nan, nan],
       [nan, nan, nan, …, nan, nan, nan],
       …,
       [nan, nan, nan, …, nan, nan, nan],
       [nan, nan, nan, …, nan, nan, nan],
       [nan, nan, nan, …, nan, nan, nan]])
```

[22]:
```
# Now let's do the Tremaine Weinberg step. Defining slits of 5 arcsec.
# The programme will align the axes using the PA of the line of nodes as␣
 ↪provided.
# The warning is just about nan and 0's being used in the division.
mydisc.tremaine_weinberg(slit_width=5.0, map_name="MUSE")
```

[23]:
```
# And you can now look at the result
print("Slicings: ", mydisc.slicings)
# Looking at the slicings
print("MUSE Slicing", mydisc.slicings['MUSE'])
# and its content
print("Yedges = ", mydisc.slicings['MUSE'].yedges)
print("Nslits?: ", mydisc.slicings['MUSE'].nslits)
print("Omega sinus(inclin) of TW method", mydisc.slicings['MUSE'].Omsini_tw)
```

```
Slicings:  {'MUSE': <pydisc.disc_data.Slicing object at 0x7f1ae5005c90>}
MUSE Slicing <pydisc.disc_data.Slicing object at 0x7f1ae5005c90>
Yedges =  [-444.6000061  -439.64372195 -434.6874378  -429.73115364 -424.77486949
 -419.81858534 -414.86230119 -409.90601703 -404.94973288 -399.99344873
 -395.03716457 -390.08088042 -385.12459627 -380.16831211 -375.21202796
 -370.25574381 -365.29945966 -360.3431755  -355.38689135 -350.4306072
 -345.47432304 -340.51803889 -335.56175474 -330.60547058 -325.64918643
 -320.69290228 -315.73661813 -310.78033397 -305.82404982 -300.86776567
 -295.91148151 -290.95519736 -285.99891321 -281.04262905 -276.0863449
 -271.13006075 -266.1737766  -261.21749244 -256.26120829 -251.30492414
 -246.34863998 -241.39235583 -236.43607168 -231.47978752 -226.52350337
 -221.56721922 -216.61093507 -211.65465091 -206.69836676 -201.74208261
 -196.78579845 -191.8295143  -186.87323015 -181.91694599 -176.96066184
 -172.00437769 -167.04809354 -162.09180938 -157.13552523 -152.17924108
 -147.22295692 -142.26667277 -137.31038862 -132.35410446 -127.39782031
 -122.44153616 -117.48525201 -112.52896785 -107.5726837  -102.61639955
  -97.66011539  -92.70383124  -87.74754709  -82.79126293  -77.83497878
  -72.87869463  -67.92241048  -62.96612632  -58.00984217  -53.05355802
  -48.09727386  -43.14098971  -38.18470556  -33.2284214   -28.27213725
  -23.3158531   -18.35956895  -13.40328479   -8.44700064   -3.49071649
    1.46556767    6.42185182   11.37813597   16.33442013   21.29070428
   26.24698843   31.20327259   36.15955674   41.11584089   46.07212504
   51.0284092    55.98469335   60.9409775    65.89726166   70.85354581
   75.80982996   80.76611412   85.72239827   90.67868242   95.63496657
  100.59125073  105.54753488  110.50381903  115.46010319  120.41638734
  125.37267149  130.32895565  135.2852398   140.24152395  145.1978081
  150.15409226  155.11037641  160.06666056  165.02294472  169.97922887
  174.93551302  179.89179718  184.84808133  189.80436548  194.76064963
  199.71693379  204.67321794  209.62950209  214.58578625  219.5420704
  224.49835455  229.45463871  234.41092286  239.36720701  244.32349116
  249.27977532  254.23605947  259.19234362  264.14862778  269.10491193
  274.06119608  279.01748024  283.97376439  288.93004854  293.88633269
  298.84261685  303.798901    308.75518515  313.71146931  318.66775346
  323.62403761  328.58032177  333.53660592  338.49289007  343.44917422
  348.40545838  353.36174253  358.31802668  363.27431084  368.23059499
  373.18687914  378.1431633   383.09944745  388.0557316   393.01201575
  397.96829991  402.92458406  407.88086821  412.83715237  417.79343652
  422.74972067  427.70600483  432.66228898  437.61857313  442.57485728
  447.53114144  452.48742559  457.44370974  462.3999939 ]
Nslits?:  183
Omega sinus(inclin) of TW method [ 1.36584209e+00  4.42720385e-01
8.60890487e-01  8.91916778e-01
  1.01887397e+00  1.12677455e+00  1.29922950e+00  1.52026779e+00
  1.73438155e+00  1.45895083e+00  1.65288730e+00  2.27774676e+00
  2.49067530e+00  2.39708406e+00  2.59152431e+00  2.67307449e+00
  2.51571531e+00  2.45459896e+00  2.57868342e+00  2.42628322e+00
  2.48510231e+00  3.03761616e+00  3.40392745e+00  2.80401774e+00
  2.14804939e+00  2.32846471e+00  1.99998962e+00  2.75068621e+00
```

```
-1.97921098e+00 -6.34668953e-01 -2.49995830e+00  2.77421022e+00
  2.56413857e+00  2.85287813e+00  3.06649097e+00  2.98899033e+00
  3.35466345e+00  3.43258449e+00  2.96293350e+00  2.85021431e+00
  2.83302733e+00  2.73579571e+00  2.60612296e+00  2.52481519e+00
  2.64092313e+00  2.57612229e+00  2.45920579e+00  2.25108140e+00
  2.40738896e+00  2.40974388e+00  2.28052832e+00  2.33244595e+00
  2.01928247e+00  2.01247901e+00  1.91957973e+00  1.71610686e+00
  1.18028023e+00  1.06521558e+00  5.81302221e-01  5.11827588e-01
  7.36411115e-01  3.42950758e-01  2.36721964e-01 -4.03374252e-02
 -4.04870330e-01 -3.53295124e-01 -3.85780530e-01 -7.00057722e-01
 -1.41751654e+00 -1.11581371e+00 -1.32442684e+00 -2.10928989e+00
 -2.81208031e+00 -3.52903977e+00 -3.98986526e+00 -4.24971034e+00
 -4.96761155e+00 -5.61127947e+00 -6.02145829e+00 -7.46863751e+00
 -8.29641274e+00 -8.65874016e+00 -8.84767114e+00 -9.67328432e+00
 -1.22497347e+01 -1.65898614e+01 -2.46691478e+01 -4.54140877e+01
 -1.42459165e+02 -5.78361666e+02 -3.87568719e+02 -1.01130674e+04
  4.07281679e+02  1.74086535e+02  1.50151246e+02  1.13549267e+02
  8.17687141e+01  5.60569970e+01  4.27795789e+01  3.71220851e+01
  3.33179307e+01  2.92341052e+01  2.69925897e+01  2.58567647e+01
  2.32625898e+01  2.22769132e+01  2.06510298e+01  1.94715622e+01
  1.93705886e+01  1.86270521e+01  1.78387325e+01  1.68316823e+01
  1.66659049e+01  1.63120067e+01  1.58888745e+01  1.57033298e+01
  1.55177763e+01  1.53407434e+01  1.50775547e+01  1.46864931e+01
  1.37264815e+01 -6.05786038e+00  1.16845907e+01  1.40030837e+01
  1.38997935e+01  1.18438898e+00  6.26674121e+02 -4.94925706e+00
  1.33072565e+01  1.27941378e+01  1.29216932e+01  1.32224450e+01
  1.31936474e+01  1.32545558e+01  1.31769379e+01  1.33576161e+01
  1.33932078e+01  1.35710881e+01  1.33022779e+01  1.27389872e+01
  1.26534863e+01  1.26150313e+01  1.26922022e+01  1.18612911e+01
  1.13335792e+01  1.15090556e+01  1.15805698e+01  1.16667943e+01
  1.16828641e+01  1.15248711e+01  1.11902472e+01  1.12048364e+01
  1.13607754e+01  1.18114023e+01  1.19244147e+01  1.17988787e+01
  1.19023758e+01  1.22709406e+01  1.27383856e+01  1.35187529e+01
  1.37745123e+01  1.35294340e+01  1.32234530e+01  1.30755479e+01
  1.33255980e+01  1.33612057e+01  1.35345434e+01  1.36788919e+01
  1.38949421e+01  1.39983729e+01  1.41118833e+01  1.40846612e+01
  1.43335900e+01  1.42065604e+01  1.40220400e+01  1.45060338e+01
  1.52741243e+01  1.59098822e+01  1.67997014e+01  1.32288140e+01
  9.04501770e+00  7.97708897e+00  6.26648599e+00]
```

2.2 3.2 Torques

Now let's consider the other class inheriting from the GalacticDisc class, namely: GalacticTorque, which itself uses TorqueMap(s).

[24]:
```python
from pydisc.torques import GalacticTorque
```

```
n6951folder = "/soft/python/pytorque/examples/data_6951/"
mass6951 = pyfits.getdata(n6951folder+"r6951nicmos_f160w.fits")
gas6951 = pyfits.getdata(n6951folder+"co21-un-2sigma-m0.fits")
gas6951 = gas6951.reshape(gas6951.shape[0]*gas6951.shape[1], gas6951.shape[2])
vc6951 = "rot-co21un-01.tex"
```

[25]:
```
t51 = GalacticTorque(vcfile_name=n6951folder+vc6951, vcfile_type="ROTCUR",␣
↪dtypemass='massd',
                    datamass=mass6951, datacomp=gas6951, Xcenmass=178.0,␣
↪Ycenmass=198.0,
                    Xcencomp=148.0, Ycencomp=123.0, inclination=41.5,␣
↪distance=35.0,
                    PA_nodes=138.7, pixel_scalecomp=0.1, pixel_scalemass=0.025)
```

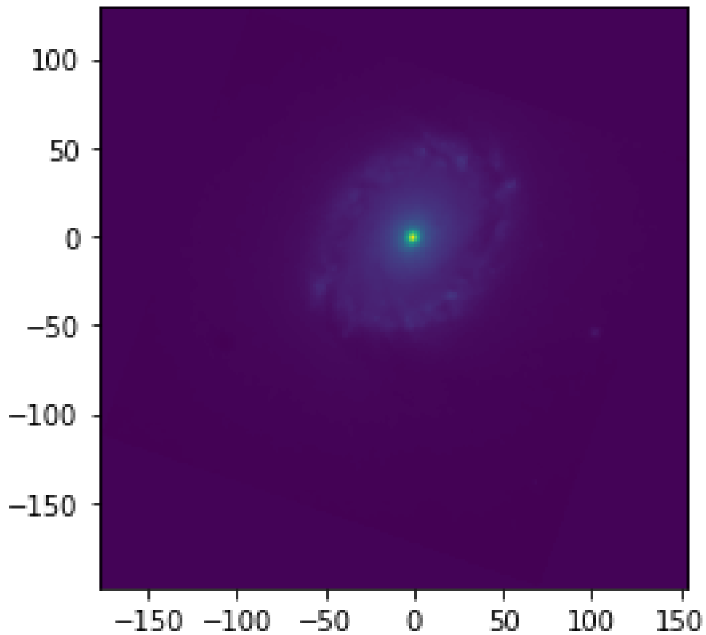
```
WARNING: X or Y not provided. Using Pixel XY grid.
INFO: attaching map mass
WARNING: X or Y not provided. Using Pixel XY grid.
INFO: attaching map comp
INFO: Adding the provided Vc file
INFO[match_datamaps]: Creating the first map massgrid and attaching first
datamap mass01
INFO[match_datamaps]: attaching the datamap dcomp01 to map massgrid
INFO: attaching map massgrid
INFO[match_comp_mass]: new map is massgrid
```

[26]:
```
t51.maps['mass'].dmaps
```

[26]:
```
{'mass01': <pydisc.disc_data.DataMap at 0x7f1ab06a0b10>}
```

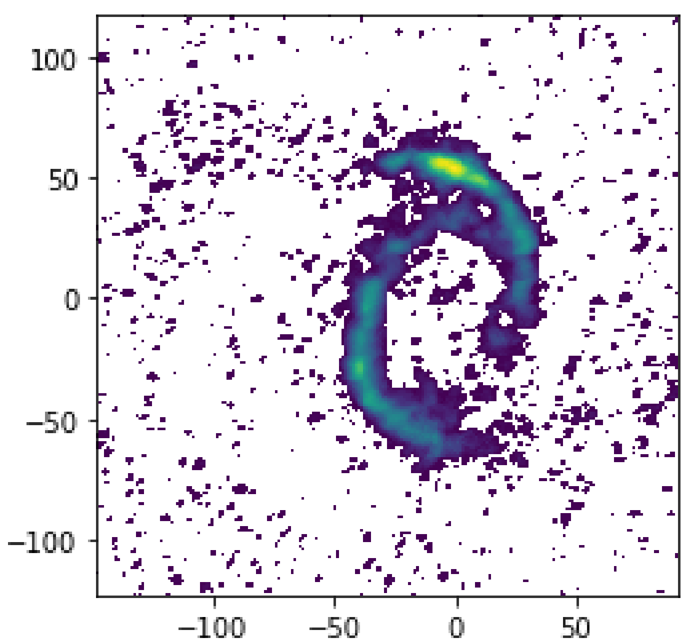
[27]:
```
from matplotlib import pyplot as plt
plt.imshow(t51.maps['mass'].dmaps['mass01'].data, extent=t51.maps['mass'].
↪XY_extent)
```

[27]:
```
<matplotlib.image.AxesImage at 0x7f1ab0275ad0>
```



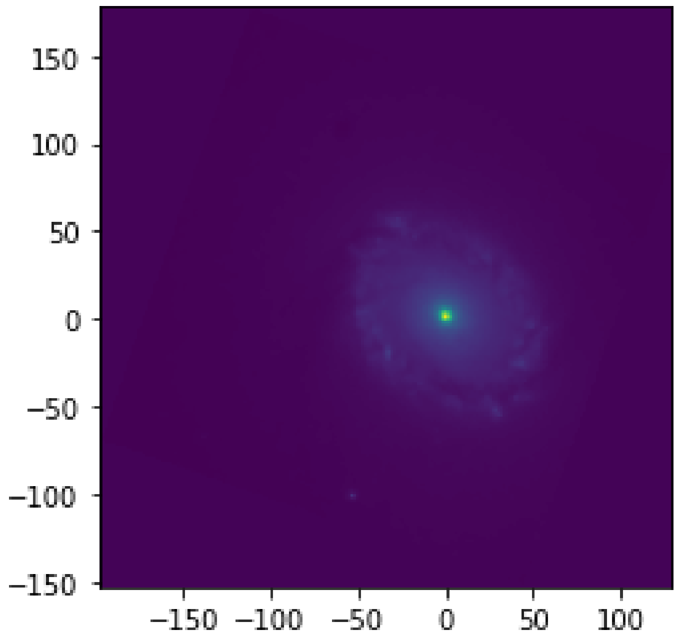
[28]:
```
plt.imshow(t51.maps['comp'].dmaps['comp01'].data, extent=t51.maps['comp'].
 ↪XY_extent)
```

[28]: <matplotlib.image.AxesImage at 0x7f1ab011dfd0>



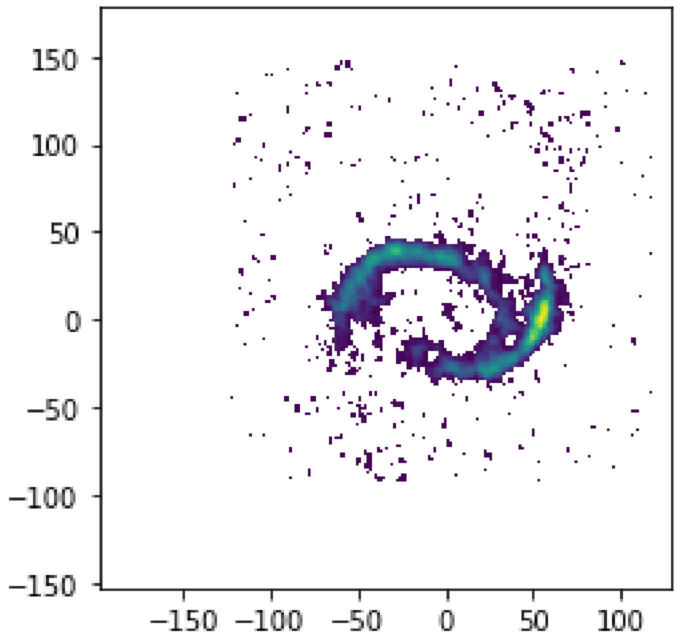
[29]:
```
# but note that these maps are now on the same grid in the massgrid map
plt.imshow(t51.maps['massgrid'].dmaps['dmass'].data, extent=t51.
 ↪maps['massgrid'].XY_extent)
```

[29]: <matplotlib.image.AxesImage at 0x7f1ab008d750>



[30]:
```
plt.imshow(t51.maps['massgrid'].dmaps['dcomp'].data, extent=t51.
 ↪maps['massgrid'].XY_extent)
```

[30]: <matplotlib.image.AxesImage at 0x7f1aafffe350>



[31]:
```
# Now running the torques
t51.run_torques()
```

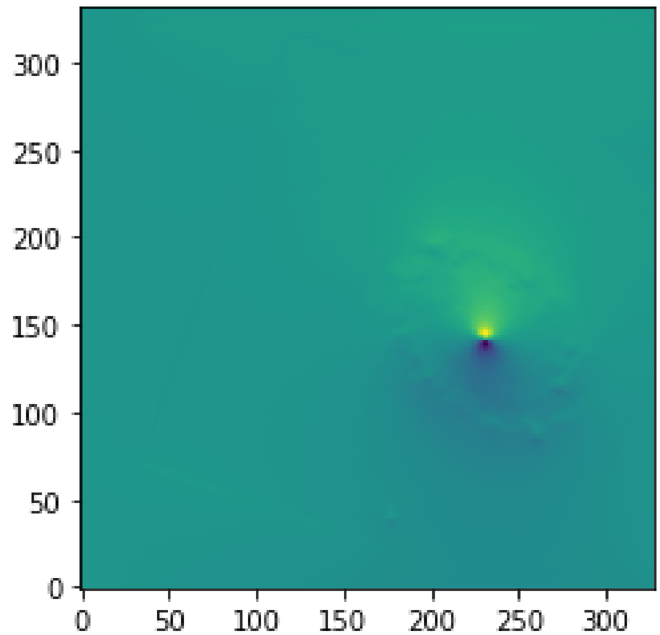
```
Deriving the radial profile …


/home/soft/python/pydisc/src/pydisc/gravpot_functions.py:116: RuntimeWarning:
invalid value encountered in greater
  goodw = (weights > 0.).ravel()
```

[32]:
```
t51.tmaps
```

[32]:
```
{'Torq01': <pydisc.torques.TorqueMap at 0x7f1ab00a07d0>}
```

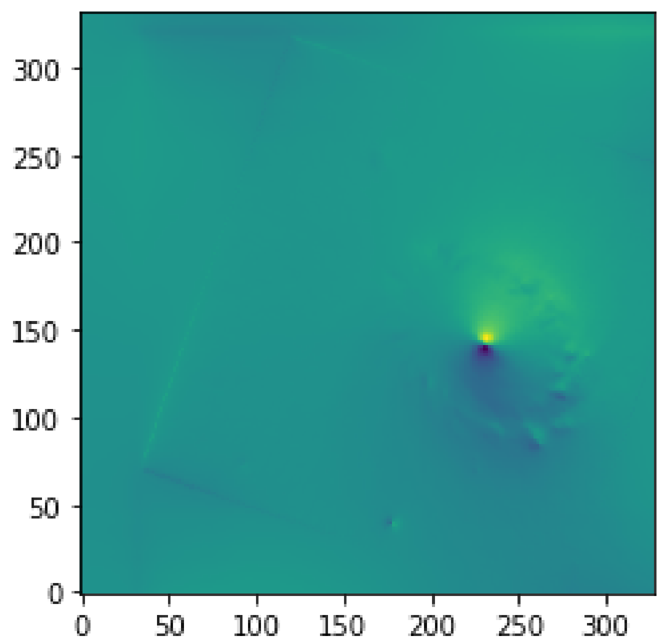
[33]:
```
plt.imshow(t51.tmaps['Torq01'].Fx)
```

[33]:
```
<matplotlib.image.AxesImage at 0x7f1aafff1b90>
```



[34]:
```
plt.imshow(t51.tmaps['Torq01'].torque_map)
```

[34]: <matplotlib.image.AxesImage at 0x7f1aaff5fcd0>



[35]:
```python
# and now for 4579
from astropy.io import fits as pyfits
from pydisc.misc_io import extract_fits
from pydisc.torques import GalacticTorque
from astropy.table import Table
folderd = '/home/science/PHANGS/Test_Packages/pydisc/for_eric/'
dCO, hCO, stepCO = extract_fits(folderd + "/NGC4579_CO.fits")
dmass, hmass, stepmass = extract_fits(folderd + "NGC4579.stellar_mass.fits")

folder_rot = '/home/science/PHANGS/ALMA/'
rot = Table.read(folder_rot + "RC_master_table_Nov2019_apy.txt", format='ascii')
RCO = rot['Radius[kpc]'].data
VCO = rot['ngc4579_Vrot'].data
RCO_arcsec = RCO * 1000 / 95.12323059

dist = 19.8
inclin = 36.0
PA =95
t4579 = GalacticTorque(datavel=VCO, Rvel=RCO_arcsec, dtypemass="massd",
                       datamass=dmass, datacomp=dCO, Xcenmass=374.4, Ycenmass=406.
 ↪3,
                       Xcencomp=233.28, Ycencomp=224.07, inclination=inclin,␣
 ↪distance=dist,
                       PAnodes=PA, pixel_scalecomp=stepCO[0],␣
 ↪pixel_scalemass=stepmass[0])
```

```
Opening the Input image:
/home/science/PHANGS/Test_Packages/pydisc/for_eric//NGC4579_CO.fits
Read pixel size of Main Image = [0.5 0.5]
Opening the Input image:
/home/science/PHANGS/Test_Packages/pydisc/for_eric/NGC4579.stellar_mass.fits
Read pixel size of Main Image = [0.7499988 0.7499988]
WARNING: X or Y not provided. Using Pixel XY grid.
INFO: attaching map mass
WARNING: X or Y not provided. Using Pixel XY grid.
INFO: attaching map comp
INFO: attaching prof vel
```

```
WARNING: VerifyWarning: Invalid value for 'BLANK' keyword in header:
-0.0350246446927 The 'BLANK' keyword must be an integer.  It will be ignored in
the meantime. [astropy.io.fits.hdu.image]
WARNING: VerifyWarning: Invalid 'BLANK' keyword in header.  The 'BLANK' keyword
is only applicable to integer data, and will be ignored in this HDU.
[astropy.io.fits.hdu.image]
WARNING: FITSFixedWarning: The WCS transformation has more axes (3) than the
image it is associated with (2) [astropy.wcs.wcs]
WARNING: FITSFixedWarning: 'obsfix' made the change 'Set OBSGEO-L to
-67.754929 from OBSGEO-[XYZ].
```

```
Set OBSGEO-B to   -23.022886 from OBSGEO-[XYZ].
Set OBSGEO-H to     5053.796 from OBSGEO-[XYZ]'. [astropy.wcs.wcs]
```

```
INFO[match_datamaps]: Creating the first map massgrid and attaching first
datamap mass01
INFO[match_datamaps]: attaching the datamap dcomp01 to map massgrid
INFO: attaching map massgrid
INFO[match_comp_mass]: new map is massgrid
```

[36]:
```
t4579.run_torques()
```

```
Deriving the radial profile …
```

```
/home/soft/python/pydisc/src/pydisc/fit_functions.py:45: RuntimeWarning: invalid
value encountered in greater
  sel_good = (flux > 0.)
```

[37]:
```
t4579.tmaps['Torq01']
```

[37]:
```
<pydisc.torques.TorqueMap at 0x7f1ae4025a10>
```

[40]:
```
plt.imshow(t4579.tmaps['Torq01'].torque_map, vmin=-1, vmax=1, extent=t4579.
 ↪tmaps['Torq01'].XYpc_extent)
```

[40]:
```
<matplotlib.image.AxesImage at 0x7f1aadd939d0>
```