Manual of plotastrodata

Yusuke Aso

1 Read Data

The data class AstroData can take a fits file.

```
from plotastrodata.analysis_utils import AstroData

d = AstroData(fitsimage='file_name.fits')
```

AstroData can take more arguments, such as `Tb` and `sigma`. `Tb=True` means the data values will be converted from flux densities in the unit of Jy beam^{-1} to brightness temperatures in the unit of K. `sigma` specifies how to measure the noise level of the data values: for example, the default option of 'hist' means to use the histgram of the data values.

The data class AstroFrame is necessary to form the AstroData instance in a useful format. AstroFrame can take quantities related to coordinate ranges.

```
from plotastrodata.analysis_utils import AstroFrame

f = AstroFrame(vmin=-5.0, vmax=5.0, vsys=2.0,
               center='00h00m00s 00d00m00s', rmax=3.0)
```

`vmin`, `vmax`, and `vsys` are in the unit of km s^{-1}. `rmax` is in the unit of arcsec. Instead of `rmax`, other arguments (`xmin`, `xmax`, `ymin`, `ymax`, `xoff`, `yoff`) can be used to adjust the x and y ranges in more detail.

Forming the AstroData instance needs the following command.

```
f.read(d)
```

After this command, the AstroData instance has useful attributes in the format of numpy array: a 1D array of `d.x` (as well as `d.y` and `d.v`), a 2D or 3D array of `d.data`, a 1D array of `d.beam`, and a float of `d.sigma`. `d.x` and `d.y` are the relative coordinates in the unit of arcsec from the given `center`. Similarly, `d.v` is the relative coordinate in the unit of km s^{-1} from the given `vsys`. `d.beam` is the beam components `array([bmaj, bmin, bpa])`, where `bmaj` and `bmin` are in the unit of arcsec, while `bpa` is in the unit of degree. When `Tb=True` above, the data values are converted to the brightness temperature and stored as `d.data`. `d.sigma` is the noise level measured in the way specified above in the same unit as `d.data`.

2 Analyze Data

AstroData also has handy methods to analyze the 2D/3D data. For example, the following method can be used to deproject the 2D/3D data with a given position angle (P.A.) and inclination angle.

```
d.deproject(pa=45, incl=45)
```

`pa` is the position angle from the north to the east in the unit of degree. `incl` is the inclination angle; `incl=0` means the face-on configuration and thus no deprojection. This command replaces `d.data` and `d.beam` with the deprojected data and the deprojected beam, respectively.

After `d.data` is updated, the data can be exported as a fits file by the following command.

```
d.writetofits(fitsimage='new_file_name.fits')
```

The output fits file reuses the header components of the fits file used to make the AstroData instance (`'file_name.fits'` above); some header components are updated properly, such as `NAXIS1`.

3 Plot Data

The class PlotAstroData can take an AstroData instance through the method of `d.todict()`.

```
from plotastrodata.plot_utils import PlotAstroData

p = PlotAstroData(rmax=3.0)
p.add_color(**d.todict())
p.add_scalebar(length=50 / 140, label='50 au')
p.set_axis()
p.savefig('figure_name.png')
```

These commands make a color map using the AstroData instance `d`. PlotAstroData can take the same arguments as AstroFrame to define the plotting ranges; particularly `rmax` is necessary. The method `p.add_color()` can take a fits file directly instead of the AstroData instance, as `p.add_color(fitsimage='file_name.fits')`. The command `p.add_scalebar()` can be omitted if the map does not need to show a scale bar. The command `p.set_axis()` (or `p.set_axis_radec()`) is necessary even without any argument. More detailed usage can be found in the `example.py` file and `https://plotastrodata.readthedocs.io/en/latest/#`.