Spectral_image class guide

isabelpostmes

January 2021

1 Guide

Lets talk through the Spectral_image class. We start by loading a spectral image, saved in a .dm3 or .dm4 file through:

```
>>> im = Spectral_image.load_data('path/to/dmfile.dm4')
```

This calls on an alternative constructor, in which the data from the dm-file is loaded, and plugged into the regular constructor. In this function, the loading package `ncempy.io.dm` is used, more info here.

```
81      @classmethod
82      def load_data(cls, path_to_dmfile):
83          """
84          INPUT:
85              path_to_dmfile: str, path to spectral image file (.
                    dm3 or .dm4 extension)
86          OUTPUT:
87              image -- Spectral_image, object of Spectral_image
                    class containing the data of the dm-file
88          """
89          dmfile = dm.fileDM(path_to_dmfile).getDataset(0)
90          data = np.swapaxes(np.swapaxes(dmfile['data'], 0,1),
                1,2)
91          ddeltaE = dmfile['pixelSize'][0]
92          pixelsize = np.array(dmfile['pixelSize'][1:])
93          energyUnit = dmfile['pixelUnit'][0]
94          ddeltaE *= cls.get_prefix(energyUnit, 'eV')
95          pixelUnit = dmfile['pixelUnit'][1]
96          pixelsize *= cls.get_prefix(pixelUnit, 'm')
97          image = cls(data, ddeltaE, pixelsize = pixelsize)
98          return image
```

Furthermore, we see the `cls.get_prefix()`, which is a small function which recognises the prefix in a unit and transfers it to a numerical value (e.g. 1E3 for k), see lines 870-916 in the complete code. Furthermore, the general constructor is called upon with

`cls(data, ddeltaE, pixelsize = pixelsize)`.

The spectral image class starts by defining some constant variables, both class related and physical. The class constructor takes in at least the data of the spectral image, `data`, and the broadness of the energy loss bins, `deltadeltaE`. Other metadata can be given if known.

```
41 class Spectral_image():

53     def __init__(self, data, deltadeltaE, pixelsize = None,
           beam_energy = None, collection_angle = None, name = None
           ):
54         self.data = data
55         self.ddeltaE = deltadeltaE
56         self.determine_deltaE()
57         if pixelsize is not None:
58             self.pixelsize = pixelsize
59         self.calc_axes()
60         if beam_energy is not None:
61             self.beam_energy = beam_energy
62         if collection_angle is not None:
63             self.collection_angle = collection_angle
64         if name is not None:
65             self.name = name
```

2 Complete code