Spectral_image class guide

isabelpostmes

January 2021

1 Guide

1.1 Loading data

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, which you can find in the complete code. 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.

Also, the delta_E axis, that is the energy-loss axis is determined by `self.determine_deltaE()`, based upon the broadness of the energy-loss bins and the index at which the average of all spectra in the image has it maximum. The definition of `determine_deltaE()` can be found at line 101 in the complete code. The image axes are determined by `self.calc_axes()`, and output either index arrays, or, if the pixel size is defined, the spacings array in meters. The definition of `calc_axes()` can be found at line 116 in the complete code.

The definitions of some other properties, such as `im.l`, `im.image_shape`, and `im.shape` can be found in the complete code from line 69 onwards.

Furthermore, there are some retrieving functions, such as `im.get_data()`, `im.get_deltaE()`, `im.get_metadata`, and `get_pixel_signal`, which should be quite self-explainatory, but whose definitions can be found in the complete code from line 124 onwards.

```
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
```

1.2 Preparing data

Now that we have loaded the data, we can perform some operations on the data of the image before starting any calculations, if wished. For example, if we wish to cut the image

2 Complete code