basic_usage

February 13, 2023

[ ]:
```
#2021/04/05 Tutorial on basic functions, first attempt of using VScode

%reset
%pylab

import os

%load_ext autoreload
%autoreload 2
```

```
Using matplotlib backend: Qt5Agg
Populating the interactive namespace from numpy and matplotlib
```

0.1 Overview

`PySurf` library consists in a set of classes and functions reapresenting 2D data and operations on them.

[ ]:
```
from pySurf.data2D_class import Data2D
```

[ ]:
```
%matplotlib inline
```

The main class representing 2D data with `x` and `y` axis is `Data2D` object in `pySurf.data2D_class`. `Data2D` can be initialized by providing a matrix of 2-dimensional data and (optionally) `x` and `y` coordinates. Other options can be passed as well.

The object interface is built on top of a function library in module `pySurf.data2D`: for almnost each method there is a corresponding function that can be called with something like `pySurf.data2D.function(data, x, y, ..)`.

Similarly, routines operating on profiles (y as a function of x as couples of vector x and y), are contained in class `pyProfile.Profile` and `pyProfile.profile` which have in many points interfaces similar to modules in `pySurf`.

Here we will focus on `Data2D` object interface.

A first way to initialize such an object is by passing directly 2D data, (optionally) coordinates and options.

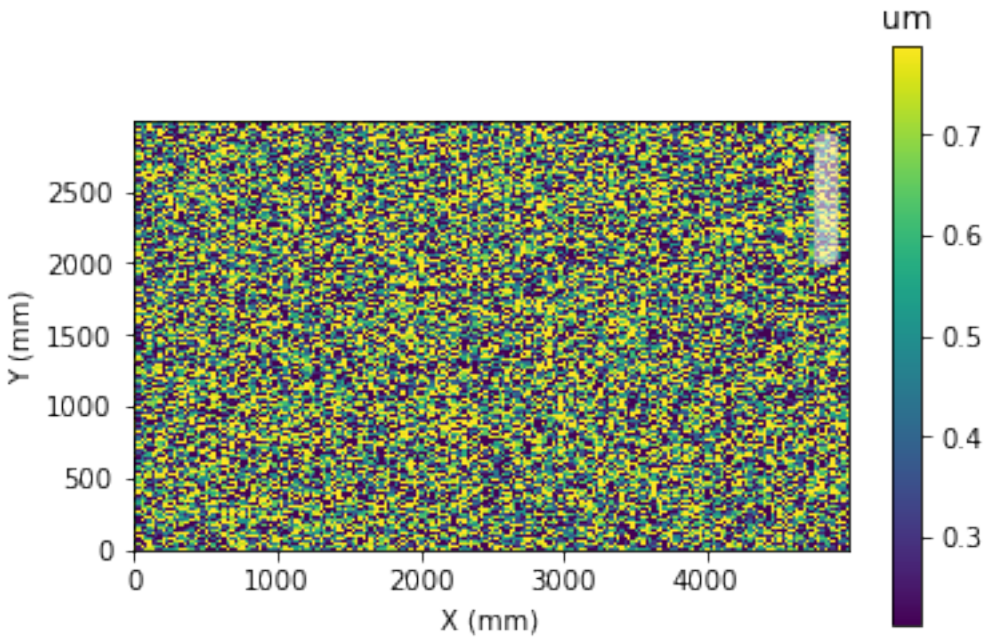
[ ]:
```
nx = 200
ny = 300
```

```
data = np.random.random(nx*ny).reshape(ny,nx)
x = np.arange(nx)*25
y = np.arange(ny)*10

D = Data2D(data,x,y,units=['mm','mm','um'])
#D.plot()
```

[ ]:
```
D.plot()
```

[ ]:
```
<matplotlib.image.AxesImage at 0x1d76a893278>
```



Data and coordinates can be obtained back in any moment simply calling the object:

[ ]:
```
dd, xx, yy = D()
```

[ ]:
```
len(xx)
```

[ ]:
```
200
```

[ ]:
```
dd, xx, yy = D()

print(dd.shape,xx.shape,yy.shape)
print("Equal?", np.all(dd == data), np.all(x == xx), np.all(y ==yy))
```

```
(300, 200) (200,) (300,)
Equal? True True True
```

Functions for reading common formats of 2D data are collected in `pySurf.readers` module. The structure and interface of readers is described elsewhere, a reader is essentially a function able to obtain `data, x, y` from a data file, however if the interface is correctly implemented, a reader from `pySurf.readers.instrumentReader` can be passed as argument to at object creation. In this case, additional information are automatically added to the object.

[ ]:
```
infolder=r'..\..\test\input_data\4D\180215_C1S06_cut'
fn = '180215_C1S01_RefSub.csv'
file = os.path.join(infolder,fn)
```

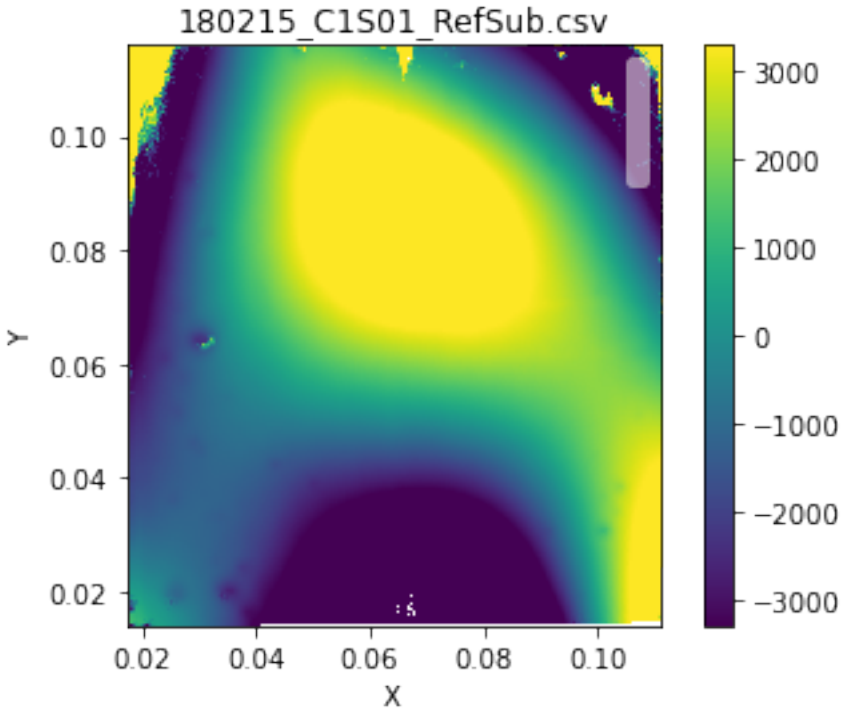
[ ]:
```
from pySurf.data2D_class import Data2D
from pySurf.instrumentReader import matrix4D_reader

D = Data2D(file, strip=True, reader = matrix4D_reader)
```

```
first argument is string, use it as filename
```

[ ]:
```
D.plot()
```

[ ]:
```
<matplotlib.image.AxesImage at 0x1d771afd6a0>
```



0.2 Data manipulation functions

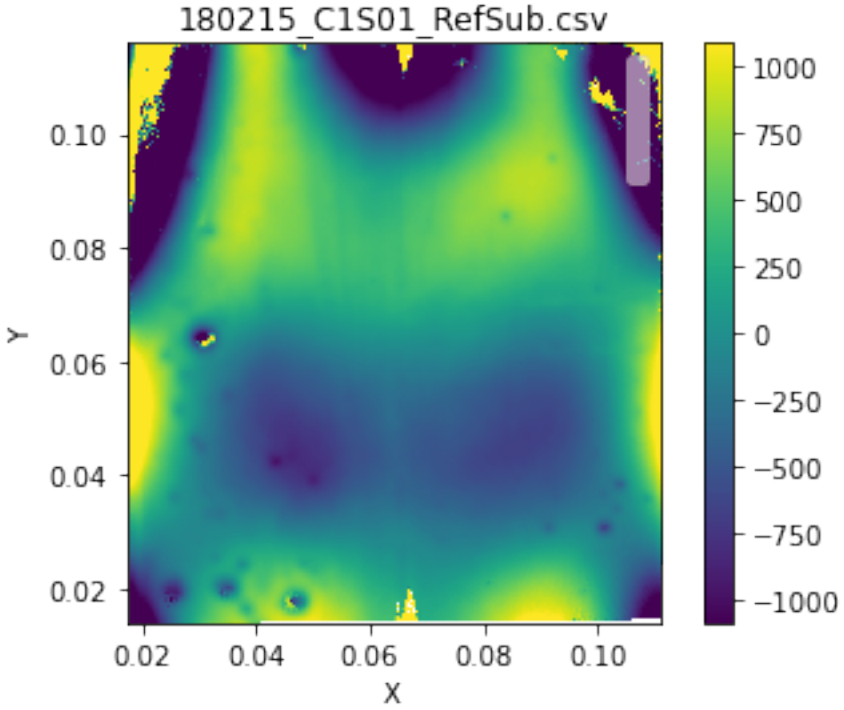
Basic operations like cropping or leveling can be applied by means of corresponding methods. Information about methods and options can be obtained by means of usual Python introspection

methods.

Here some example:

[ ]:
```
D2 = D.level((4,2))
D2.plot()  #Level 4 legendre along `x` and 2 along `y`.
```

[ ]:
```
<matplotlib.image.AxesImage at 0x1d76c775898>
```

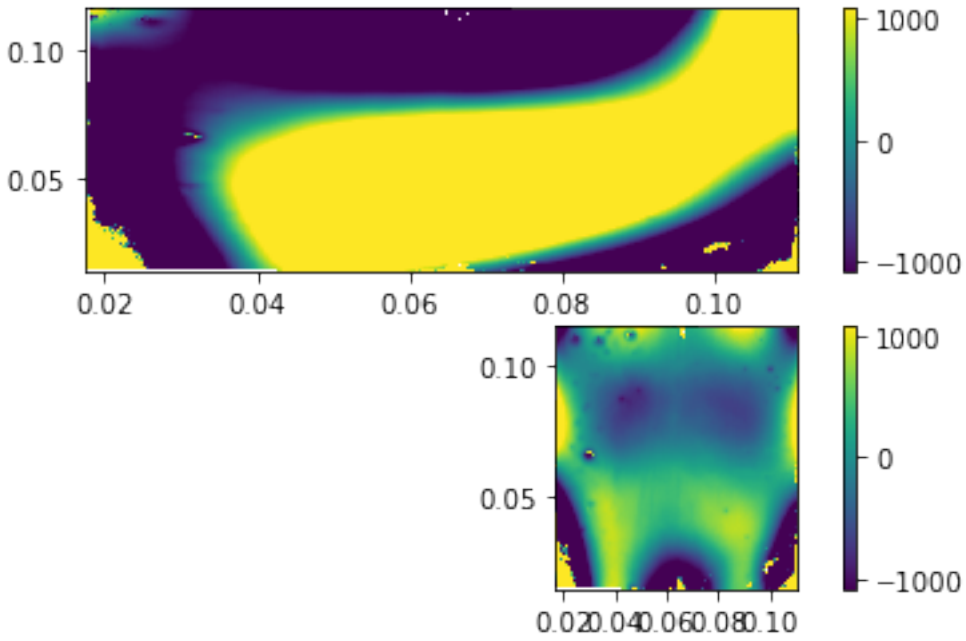


`plotting` module contains commodity functions for plotting of data and comparisons, we use `plotting.multiplots.compare_images` to compare the original data with the modified version.

[ ]:
```
from pySurf.data2D import compare_2images

compare_2images(D.data,D2.data,D.x,D.y)
```

[ ]:
```
[<AxesSubplot:>, <AxesSubplot:>]
```



[ ]:
```
D2 = D.level((4,2))

for d in compare_images(D,D2):
    d.plot()
```

[ ]:
```
from pySurf.psd2d import calculatePSD
from pySurf.psd2d import psd2d, plot_psd2d,avgpsd2d
from pySurf.data2D import crop_data
```