Peer-review of assignment 4 for *INF3331-ererikse*

Reviewer 1, clasjog, clasjog@ifi.uio.no
Reviewer 2, shashaj, shashaj@student.matnat.uio.no
Reviewer 3, chriklev, chriklev@student.matnat.uio.no

October 9, 2018

1 Review

System specifications used for review:

- Python 3.6.6
- Ubuntu 18.04.1 LTS

Assignment 4.1

The code runs smoothly and works as expected.
The code is very well documented without being "over documented". The main method could use explanations for its arguments.
The code is very organized and easy to read with intuitive variable names.

Assignment 4.2

Just as in the previous part, the code is well documented and easy to read with intuitive variable names. However it is word for word the exact same code except two places where you do the same with slightly different syntax. The point of this task was to vectorize the code using numpy by applying algorithms to an array of variables instead of one element at a time. The way i did this was to create a 2D array of the initial complex numbers using numpy.meshgrid, and use numpy.less to check which elements that haven't reached a value of 2 yet in the loop. Then I use smart indexing to get the elements i want to change.

```
image = np.zeros(resolution)

realNums = np.linspace(realMin, realMax, resolution[0])
imagNums = np.linspace(imagMin, imagMax, resolution[1])
z = np.sum(np.meshgrid(realNums, imagNums * 1j), axis=0)

w = np.zeros(resolution, dtype="complex64")

for i in range(iterations):
    ind = np.less(w, 2)
    image[ind] = i
    w[ind] = w[ind]**2 + z[ind]
```

I also noticed that mandelbrot_2.py runs nearly twice as slow as mandelbrot_1.py. I don't know why, but I assume it has to do with the way you initialize the complex numbers since it is the only difference.

```
# From mandelbrot_1.py
real = row[i]
imag = col[j]
temp_matrix[i, j] = mandelbrot(complex(real, imag), max_iterations)
```

```
# From mandelbrot_2.py
temp_matrix[i, j] = mandelbrot(row[i] + 1j*col[j], max_iterations)
```

Assignment 4.3

This is te same code as earlier but with jit'ed functions, wich was exactly what you were supposed to do. It is still well written. Easy to read understand, and well documented. Also the numba is used correctly and makes the code significantly faster. One thing i want to mention is that you can use @jit(nopython=True). This will chrash the program if numba is not able to compile something. Without (nopython=True) it will just skip the part it couldn't compile and run the script as normal without telling you. This wasn't a problem here, but i wanted to mention it. :)

Assignment 4.4

I'm taking INF3331 and don't know how Cython works, but i tried to build and run the code and got this:

```
\$ python3 mandelbrot_4.py
Program started
Traceback (most recent call last):
  File "mandelbrot_4.py", line 13, in <module>
    main(-2.0, 0.5, -1.25, 1.25, 1000, 1000, "cython_image", 80)
  File "mandelbrot_4.py", line 6, in main
    mandelbrot_set(xmin, xmax, ymin, ymax, width, height, max_iterations, image_name)
  File "mandelbrot_cython.pyx", line 20, in mandelbrot_cython.mandelbrot_set
    cpdef mandelbrot_set(double xmin, double xmax, double ymin, double ymax, int width, int height, int max
  File "mandelbrot_cython.pyx", line 25, in mandelbrot_cython.mandelbrot_set
    int[:,:] mandelbrot_matrix = np.empty((width,height), np.int)
ValueError: Buffer dtype mismatch, expected 'int' but got 'long'
```

I assume this is because i didn't do the setup properly, but i really couldn't get it to work. I guess this error would occur if width or height are long instead of int.

Assignment 4.5

The code is well organized and fairly easy to read. It is not necessary, but i did miss a few inline comments to explain what part of the code did what. Everything works as it should and all the parts of the task are answered.

Assignment 4.6

As earlier the code is organized, easy to read and well documented. The setup file works, but does not install the "compute_mandelbrot" method. The test functions are efficiently written, have intuitive names and test what they are supposed to.

Assignment 4.7

The contest picture is very nice. The option to change color scales is not implemented.

Assignment 4.8

Not done.

General feedback

In general very good. The code is consistently well organized, easy to read, and well commented. The numpy implementation was not done correctly but i saw you explained it a little bit in report_2.txt, so maybe you just didn't have time. You were also missing the "compute_mandelbrot" method. But overall, WELL DONE!