Peer-review of assignment 4 for *UiO-INF3331/INF3331-alexawii*

Reviewer 1,etiennb, etiennb@student.matnat.uio.no
Reviewer 2, sindrewi, sindrewi@student.matnat.uio.no
Reviewer 3, AddYour-UiO-username, youremail@uio.no

October 11, 2017

1 Introduction

1.1 Goal

The review should provide feedback on the solution to the student. The main goal is to *give constructive feedback and advice* on how to improve the solution. You, the peer-review team, can decide how you organise the peer-review work between you.

1.2 Guidelines

For each (coding) exercise, one should review the following points:

- Is the code **working as expected**? For non-internal functions (in particular for scripts that are run from the command-line), does the program handle invalid inputs sensibly?
- Is the code **well documented**? Are there docstrings and are the useful?
- Is the code written in **Pythonic way** [1]? Is the code easy to read? Are the variable/class/function names sensible? Do you find overuse of classes or not sufficient use of functions or classes? Are there parts of the program that are hard to understand?
- Can you find **unnecessarily complicated parts** of the program? If so, suggest an improved implementation.
- List the programming parts that are not answered.

Use (shortened) code snippets where appropriate to show how to improve the solution.

1.3 Points

The review is completed by pushing the review Latex source file (.tex files) and the PDF files to each of the reviewed repositories. The name of the files should be *feedback.tex* and *feedback.pdf* in the students assignemnt4 directory.

You will get up to 10 points for delivering the peer-reviews. Each of you should contribute to the review roughly equivalently - your team will get the same number of points[2].

2 Review

Specify the system (Python version, operating system, ...) that was used for the review. For the review I was using a windows 10 system and PyCharm IDE. Cython is compile with visual dll .

General feedback

Globally it's a good work and most the part are doing there job (small problem on numpy midpoint) You are not using comment! It's a small package and most of the function are simple but it's always a good training. A small documentation on your main integrate function is necessary. Explain the input and the output. Be careful with your import, they are very relative and I had to redo some of them to make your code work. The "_init_.py" file is very importantfor module. You pick good name for your variables and your function, it's make your code easy to read. Globally, you are giving useful name to your variables which make your code easy to read.

[1] https://www.python.org/dev/peps/pep-0020/
[2] In case a team-member does not contribute, please email simon@simula.no

Assignment 4.1

The test have a meaning full name but they are only there for numpy and default configuration. You are not supposed to call your test in your python script, instead use py.test to run it. You miss test for numba and cython. You could have write more different test to really check that your scripts are running in all condition

Assignment 4.2

Your double loop is not necessary, you can use only one:

```
def integrate(f, a, b, N):
    N = int(N)
    dx = float((b - a)) / N
    res = 0
    for i in range(1, N + 1):
        res += f(a + (i * dx)) * dx

    return res
```

I didn't succeed to run your plot_error a f function is missing and you never call your function. Still inside, the code is correct. Your plot have the good shape.

Assignment 4.3

Add a review based on section 1.2. In addition, review the following assignment specific items:

- Is numpy being used effectively (e.i. vectorization where possible)?

You don't vectorized f so numpy isn't use, more you are using classical loop but it's not needed with numpy. This package has the property to use powerful matrix computation. You should use it to improve your code speed Your midpoint implementation is not working.

Proposition of implementation

```
""" Implementation of rectangle integration """
def numpy_integrate(f, a, b, N):
    dx = (b - a) / float(N)
    A = np.linspace(a + dx , b , N)
    S=f(A)
    if isinstance(S, (int, float)):
        return S*(b-a)
    else:
        return (S[1:]*dx).sum()
```

- The function is vectorized
- No need of loop
- Take care of the constant function

Unfortunately, your report doesn'tshow good result: numpy is 40 times faster than normal implementation

Assignment 4.4

You're numba implementation is nice but there are no test to check it but the test.py who look like more a profiler. So you should implement test to check if all your functions are working. You are using the no python mode correctly witch give really fast result. The code is written in a Pythonic way because you are using good variable name and it's not too complex. Some documentation about your use of numba would have been appreciated

In your report, you show well that numba need to compile file and spend some time to do it at the begining of the fuction. That make it slower than the other method for small n.

Assignment 4.5

You're cython implementation is good and I don't see any improvement. You could had some comment but I don't think it's necessary because you are using clear naming.

Cython is suppose to perform as well as numba but your n are to small to see correct result.

Assignment 4.6

For python, numba and cython the code run as expected, not for numpy. The code is not documented.

It's look like you doesn't understand the comparison part of this assignment. You are suppose to compare the difference between midpoint and default integrator for each method. You should need a bit less N with the midpoint method. This understanding make your report showing bad informations.

Assignment 4.7

You are not using correctly setup and init to create your module

Example of _init_.py

```
from .integrator import integrate as toto
from .integrator import midpoint_integrate
from .numba_integrator import numba_integrate , numba_midpoint_integrate
from .numpy_integrator import numpy_integrate , numpy_midpoint_integrate
from .cython_integrator import cython_integrate, cython_midpoint_integrate ,integrate_f2x
```

Example of setup.py

```
from setuptools import setup
setup(
    name='integrator',
    description='An integrator package',
    packages=['integrators'],
    version="1",
)
```

Also I think that you shouldn't put your integrate function in the integrate.py script but directly in this init file.
When your module is ready, you can install it with:

```
python setup.py install
```

Assignment 4.8

You're contest is working perfectly, there is nothing to say but you don't need to integrate to 10^7 to find a good result. When you see the function, it is going to zero near 100. You just need to integrate from 0 to 100 to get the result. As you have more step, you can compute your integral and get a more accurate result.

2.1 Useful Latex snippets

Here are some sample usage of Latex.

2.1.1 Sample code

```
import sys
print "This is a sample code"
sys.exit(0)
```

2.1.2 Mathematical equation

$$2\pi > 6 \tag{1}$$