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

Kari Eriksen, KariEriksen, eriksen_kari@hotmail.com
Even Marius Nordhagen, evenmn, evenmn@student.matnat.uio.no
Bendik Steinsvåg Dalen, bendiksd, bendiksd@student.matnat.uio.no

October 12, 2017

1 Review

We used Python3 and Windows 10 for this review.

General feedback

You have a bit confusing folder structure, with, for example, some files placed mulitiple places and a lot of folders. You also haven't used that many comments describing what your code does. This made it a bit confusing to read and understand what you have done/how you were thinking.

Assignment 4.1

Your code structure here is a bit lackluster. In some of the functions you have used doubble tab as indentation, but in some of them you have not. It looks like you might have tried to seperate linear and constant functions, but in this instance it dosen't really work, and it looks distractingly weird. You could also have used some more newlines (enters(?)) after/before your functions to make some more space and make the code easier to read.

Your naming of functions and variables is good. The names makes sense, and are apropirate for their use.

Your code is unnecessarily complicated at places, and could have been simplified. For example, you could have used some for-loops to check the functions for different N:

```
for i in range(2,6):
        computed = integrate(f, a, b, 10**i)
        assert abs(computed - expected) < 1e-10
```

You should also have called the test-functions to show that they work (not just make them).

Assignment 4.2

Your code and code structure in this assignment is much better than in 4.1. It's better structured, and easier to read. It's also nice that you have implemented a "shortcut" in case of a constant function, even though this was not a part of the task. Though you don't need to use y here, $f(0)$ also works.

Your file contains some unnecessary code, which you have commented out anyways, that you could have deleted.

Your plot is fine. It's nice that you have a title and names on the axises, though a smaller timestep would have given you a smoother, and thus nicer, curve. A higher N_{min} and N_{max} would have shown the error for more relevant N's. Other ways to make the plot look better is to use something like seaborn or the grid() function.

The code would also be slightly faster if you used

```
mesh = range(Nmin, Nmax + 1, stepsize)
```

instead of appending to mesh in the for-loop.

Assignment 4.3

Your code here is fine. You have used numpy correctly, but you still have the problem with some unnecessary code. You also have imported timeit here, but not used it, for some reason. Other than that we have no new comments about your code.

Your report was good, but mabye you could have run the programs for different N, to see how differently the runtime scales.

Assignment 4.4

Your implementation of numba was good here. We don't really have any other new things to say about this task.

Assignment 4.5

Can't run the program for some reason, and are thus not able to evaluate it.

Assignment 4.6

Your code in this assinment is very good. The loop in integrator_comparison.py is clever and your use of comments are better here. The biggest problem here is how you find the midpoint. Your method gives small errors in the begining of the loop (though these errors are extremley small). A better method would be something like:

```
x = np.linspace(a, b, N + 1)
x_mid = (2*x + dx)/2
```

Or you could set x=x[1:] instead of x[0]=0. You also check if the function is linear twice in *numpy_midpoint_integrate*(). I'm guessing that's just a small mistake.

Assignment 4.7

The module works fine.

Assignment 4.8

Your code was not very accurate here. We're not able to identify exactly whats wrong with your code, and why you recive true_divide problems. Other than that, here are some ways you could improve your runtime and accuracy:

- Factorize the functions further, i.e. $1 = \frac{(\sin(x)\cdot\sin(x/3)\cdot\sin(x/5))}{(x^3)} \cdot (\frac{15}{pi})$. This will reduce workload for each step of the loop, decreasing runtime
- Set all constants outside the functons, not just π. This will also reduce the workload
- If it had worked, the midpoint method would give a more accurate answer
- Numba or cython migth allso be faster in this instance
- For these particular functions you need a very high N to even get close to the correct anwser, and yours is not that large

References