Artificial Intelligence (CO3519): Tutorial – Calendar Week 41

1. Let us jointly review the most important features of Python syntax.[1]

2. In this week's lecture, the focus was **a)** on the way in which **optimization problems** are formulated in terms of a design space (also, parameter space) and an objective space. Then, **b)** we had a closer look at the most simple case, where there is only one parameter and only one objective, so that the optimization problem reduces to finding the maximum or minimum for a function of a single variable; there, we considered three **local optimization** algorithms than can be implemented in a couple of lines of code, *e.g.*, as shown in this week's Jupyter notebook.[2] Both **a)** and **b)** will need to be revisited.

Let us focus this time on **b)**, reviewing the concepts from **local optimization**.

Note: In the example, the seaborn library is used to generate plots.[3] If your installation does not contain that library ("**import** seaborn" fails), download and install the package.[4]

3. Building on the code from this week's Jupyter notebook,[2] consider the function

$$f(x) \quad = \quad 3x^4 - 4x^3 - 6x^2.$$

Beginning with[5] an initial value x = 1.1, apply **hill climbing** (in the first of the two variants that were shown in the lecture) and **Newton's method** to numerically approximate a **local minimum** of $f(x)$. Compare the performance of the two algorithms: How well do they perform within four iterations? What are good values for α and δ in the case of the hill-climbing, and what is a good value for α in the case of Newton's method?

Note: The value of ε does not play a role here, since it only controls the termination criterion, and we now instead specify that termination must occur after four iterations.

Submit your answer to **no. 3**, preferably using a Jupyter ipynb file, via Blackboard.

Submission deadline: 30th October 2021; discussion planned for 12th November 2021. Group work by up to four people is welcome.

1 Python Software Foundation, *Python Tutorial*, https://docs.python.org/3/tutorial/, **2021**.
2 On Blackboard under Module Materials → Supplementary Material, or on the CO3519 course website.
3 See the seaborn website, https://seaborn.pydata.org/, for a presentation of its features.
4 https://anaconda.org/anaconda/seaborn
5 **Apologies for initially suggesting x = 1; this does not work for Newton at all, since $f''(1) = 0$.**