Karl M. Heuer / David E. Roberson

Autumn 2023
Exercise class:
in 302 on September 5th

Exercises for Computational Tools for Data Science (02807)

Week 2: Python recap

Exercise 1: Setup Python

Install Python on your computer and/or update it to the latest version.
Install an IDE (e.g. VS Code or Eclipse).
Install Jupyter Notebook.

Exercise 2: Tutorial on learnpython.org

Work through the Python tutorial on `www.learnpython.org`.

Exercise 3: Additional tutorial exercises from Princeton

The following website offers another Python tutorial based on a textbook:

`https://introcs.cs.princeton.edu/python/home/`

Note that parts of the tutorial are outdated since newer version of Python appeared compared to when the book appeared. Focus on the following sections together with the suggested exercises, but feel free to explore the tutorial further:

Section 1.1 (Your first program): Exercises: 1.1.1, 1.1.6.
Section 1.2 (Built-in types of data): Exercises: 1.2.2, 1.2.5., 1.2.8., 1.2.11.
Section 1.3 (Conditionals and loops): Exercises: 1.3.6, 1.3.7., 1.3.13., 1.3.27.
Section 1.4 (Arrays): Exercises: 1.4. creative 2. (longest plateau), 1.4. creative 10. (find duplicates)
Section 2.1 (Defining functions): Exercises: 2.1.3, 2.1.15.
Section 2.3 (Recursion): Exercises: 2.3. creative 3. (permutations)

Exercise 4: NumPy package

Install the NumPy package and look at `http://www.numpy.org/` for help. If you never used NumPy before read / skim through the chapter 2 of the Python Data Science Handbook.

Write a python script which reads a matrix from a file like this one:

`https://pastebin.com/XAhwshXe`

and solves the linear matrix equation $Ax = b$ where b is the last column of the input-matrix and A corresponds the other columns. You are allowed to use the solve()-function from numpy.linalg. Does the result make sense?

Optional: Execute and read through the cells in this notebook to get an understanding of vectorised operations in NumPy and why they are much faster.

Exercise 5: SciPy package

Install the SciPy package and look at `https://scipy.org/` for help.

Write a python script that reads in this list `https://pastebin.com/ENyYffaq` of points (x, y), fits/interpolates them with a polynomial of degree 3. Solve for the (real) roots of the polynomial numerically using SciPy's optimization functions (not the root function in NumPy). Does the result make sense (plot something to check)?

Exercise 6: Numba package

Install the Numba package and look at `https://numba.pydata.org` for help.

Write a simple Python function for computing the sum $\sum_{i=1}^{n} \frac{1}{i^2}$ with $n = 10000$ (the result should be around 1644), running 2000 times in a row (to make the execution time measurable).

Now use Numba and see how much quicker you can get (you can use the %timeit magic command in Jupyter Notebook for this). Can you make the Numba version faster? How much?

Exercise 7: Pandas package tutorial

Install the pandas package and look at `https://pandas.pydata.org/` for help.

Read / skim the pandas tutorials at:

`https://pandas.pydata.org/docs/getting_started/`

`https://www.kaggle.com/learn/pandas`

If you feel like you need a more in-depth introduction read through chapter 3 of Python Data Science Handbook.

- Scrap tabular data from different websites (not just wikipedia).
- Handle import and export of tabular data via csv files.
- Document / present your work via Jupyter Notebook.

Optional: Watch this PyData presentation on how to use pandas effectively. This will come in handy when you will need to work on your final project.

Exercise 8: (optional)

If you are already familiar with pandas and NumPy and felt that the above exercises were too easy, solve Exercise 1 and 2 in this notebook. Can you achieve the same or better execution speed compared to the teacher?