Individual assignment 2: Regression

Life expectancy data

Background
The World Health Organization (WHO), provide data on life expectancy and different health factors for 193 countries. One of the interesting questions that can be studied with the data is if there exist relationships between life expectancy and other the factors in the data (and if so, which factors).
The aim of this assignment is to be able to do regression in Python on a real dataset, but also understand the computational numerical methods that are at play under the hood, and analyze theoretical properties in relation to these methods.

To do

1) Download the data file **`LifeExpectancyData.csv`** from Studium. The file contains the WHO data described above. You can open the file and look at the data on your laptop before you import it. We will only use the columns *life expectancy* and *schooling* here. Import the data in Python, for example use the **`numpy`**-command **`np.genfromtxt`** here, and exclude the header line and only import the columns you are interested in:
```
dataWHO = np.genfromtxt('your_file', delimiter=',',
usecols=(3,21),skip_header=1)
```

2) Prepare the and plot the data
There are some **`nan`**-values in the data (=missing data). You can deal with missing data in different ways, but the easiest is to simply remove those lines in the data (you must remove in both columns, even if it's a **`nan`** only in one column). The simplest way to remove them in **`numpy`** is probably to use the methods **`numpy.isnan()`** and **`any()`** (if *any* of the elements in a row *is a nan*). For example, you can print all **`NaN`**'s by printing `your_data_array[~np.isnan(your_data_array).any(axis=1)]`.

   Create a new data array with the **`NaN`**-rows excluded. When it is done, plot the data, to see what it looks like. Schooling will here be the independent variable (x-axis).

3) Do polynomial fits to analyze the dependency between life expectancy and schooling. You can use the **`numpy`** polynomial package in Python (as in the lab). Try both linear and quadratic polynomial fit and plot your polynomial fits together with the data.

4) Analyze the condition numbers for the two cases in 3). Form the matrix A and A^TA for the two cases, calculate the condition numbers (you can use the built-in functions in **`numpy`**). In a worst-case scenario, how much accuracy can we lose when solving the normal equations $A^TA\hat{x} = A^Ty$?

5) **Optional (for those of you who'd like to do more…)**
   Create histograms of the residual, $r = y - A\hat{x}$. Don't use too few bins in the histogram, choose for example 200 bins.
   The difference between any data point and the regression line can be expressed with the residual. If it is correct the residual should roughly follow a normal distribution. Does it seem to be correct here?

Submit figures, results (and answers to the questions) and code in one pdf-file.

Rules

- It is mandatory to hand in the Individual lab assignment in due time
- Must be solved *individually*
- Solved *after the corresponding lab session* (at home or in computer lab).
- It is essential that you *try* to solve the problem, not that you solve it correctly. If you don't know how and what to do, hand in what you have (even if it is just a few rows of code). The task will be approved even with an incorrect or even empty solution, and you will get feedback and opportunity to improve it.

The task is linked to a computer lab. Thus, you can look back into the lab and find out how to solve the problem.