Newton-Raphson and FLOP counting

Due: Friday, March 5th, 17:00. Push the following files to your GitHub Classroom repository:

- The script test_NR.py for question 1(b).
- Your answers to question 1(a,c) and 2 in a PDF document typeset in LaTeX. It is good practice to add the source (.tex) files as well!

A discussion thread for this assignment is available on Slack. Pose your questions there before approaching the lecturer or TA.

Question 1 **15 marks**

Consider the following system of equations:

$$\begin{aligned} -x_2^2 - x_3^2 - \frac{1}{4}x_1 + 2 &= 0 \\ x_1x_2 - 4x_1x_3 - x_2 + 1 &= 0 \\ 4x_1x_2 + x_1x_3 - x_3 &= 0 \end{aligned} \tag{1}$$

There exists a solution close to $x = (7, 0, 0)^t$.

(**a**) Write a pseudo-code for Newton-Raphson iteration (an outline is included in the slides for lecture 10). Remember to

1. list the input arguments with their type and meaning;
2. clearly state the loops and conditionals;
3. state the operations in a form independent of the implementation (e.g. avoid using names of Python functions like range(...)).
4. List the output arguments with their type and meaning.

(**b**) Write a Python script that defines the system of equations (1) and its Jacobian, sets the arguments for Newton-Raphson iteration and calls the function NR.py that you can find in the `course_codes` repository. Make sure your script prints out the residual and approximate error at every Newton-Raphson step, as well as the final result. Set both tolerances to 10^{-12}. Submit your script with the name test_NR.py.

(**c**) Judging by the sequence of residuals, would you say that Newton-Raphson iteration has the same rate of convergence as Newton iteration?

Question 2 **15 marks**

A special property of lower triangular, square matrices is that the product of two lower triangular matrices is also lower triangular. To be precise, if $A_{ij} = 0$ for $j > i$ and $B_{ij} = 0$ for $j > i$ than $C = AB$ satisfies $C_{ij} = 0$ for $j > i$.

(**a**) Write a pseudo-code for a function that computes the product of two lower triangular matrices. Initialize the product as a zero matrix and compute only the elements on and below the diagonal.

(**b**) Compute the number of FLOPs it takes to complete the function on part (a) as a function of the number of rows n of the input matrices.

(**Bonus**) Your function for triangular matrix-matrix multiplication can require at most 6 times fewer FLOPs than regular matrix-matrix multiplication (asymptotically for large n). Does you algorithm require 6 times fewer FLOPs? Then the 10 bonus marks are yours. If not, find a way to make your function more efficient until it requires 6 times fewer FLOPs.

(for 10 bonus marks)