Norwegian University of Science
and Technology
Department of Mathematical
Sciences

# TMA4130
# Mathematics 4N
# Fall 2020

**Exercise set 8**

## Mandatory problems

1 We consider the equation

$$f(x) = x - e^{-x} = 0.$$

**a)** Show that this equation has a unique solution $r \in \mathbb{R}$. Show in addition that the solution satisfies the estimate $1/2 \leq x \leq 1$.

**b)** In order to compute the solution of this equation numerically, we consider the fixed point iterations

$$x_{k+1} = g(x_k),$$

where $g$ is any of the following functions:

1. $g(x) = -\ln(x)$.
2. $g(x) = e^{-x}$.
3. $g(x) = (x + e^{-x})/2$.

Test the three iterations by using (and modifying appropriately) the function `fixedpoint` from the Jupyter notes. Use $x_0 = 1/2$ as an initial value.

Which of the iterations converges? Which of the iterations shows the fastest convergence speed?

**c)** Use the fixed point theorem in order to obtain a theoretical explanation for the numerical results.

2 We want to apply fixed point iteration to the solution of the equation $\cos(x) = \frac{1}{2}\sin(x)$ using the mapping

$$g(x) = x + \cos(x) - \frac{1}{2}\sin(x)$$

with an initialisation $x_0 = 0$.

**a)** Show that the mapping $g$ satisfies the conditions for the fixed point theorem on the interval $[a, b] = [0, \pi/2]$.

**b)** Provide an estimate of the accuracy of the outcome of the method after the $5^{\text{th}}$ iteration. How many iterations will be needed to obtain a result with an error smaller than $10^{-12}$?

*Hint: Use the a-priori error estimate in the fixed point theorem.*

3 **a)** Derive the formula for Newton's method for the solution of the equation $x^n = a$ where $n$ is a natural number and $a > 0$. [1]

**b)** Use Newton's method to approximate a solution of the equation $x^4 = 4$. Use $x_0 = 1$ as starting value and perform four iterations by hand.

4 We consider the equation

$$f(x) = x^3 - x^2 - x + 1 = 0,$$

which has the two roots $r = -1$ and $r = +1$. In this exercise, we will discuss the usage of the Newton method in order to solve this equation. One can show that Newton's method for the solution of this equation converges for each initialisation $x_0 > 1$ to the root $r = 1$, and for each intitialisation $x_0 < -1$ to the root $r = -1$.

**a)** Write down the iteration scheme and perform the first three iterations with initialisation $x_0 = 2$.

**b)** Based upon the theory developed in the lecture, what convergence order do you expect for the iterations starting with $x_0 = 2$ and $x_0 = -2$, respectively?

**c)** Use and modify the function `newton` in the Jupyter notes in order to verify the convergence orders numerically.

## Additional exercises

*These additional exercises are completely optional and should not be handed in. The student assistants will not grade these problems.*

5 The *regula falsi* is a modification of the bisection method (see the Jupyter notes). However, instead of simply choosing $c_k = (a_k + b_k)/2$ as in the bisection method, in the regula falsi we join the points $(a_k, f(a_k))$ and $(b_k, f(b_k))$ by a line segment and then compute the intersection point of that line segment with the $x$-axis.

Put differently, we compute first a linear interpolation polynomial $g_k(x)$ through the points $(a_k, f(a_k))$ and $(b_k, f(b_k))$. Then we choose the point $c_k$ to be the solution of the equation $g_k(x) = 0$.

**a)** Show that this leads to the formula

$$c_k = \frac{a_k f(b_k) - b_k f(a_k)}{f(b_k) - f(a_k)}.$$

**b)** Implement the regula falsi in Python. You may build your implementation upon the function `bisection` in the Jupyter notebook.

[1] For $n = 2$ (that is, computation of square roots), this method actually goes back to antiquity and is known as "Heron's method" or the "Babylonian method". See `http://en.wikipedia.org/wiki/Methods_of_computing_square_roots#Babylonian_method`.

6 **a)** Let $g(x)$ be a continuous function with continuous derivatives on $(a, b)$ and assume that it has an inverse $g^{-1}(x)$. Show that if $r \in (a, b)$ is a fixed-point of $g(x)$, then $r$ is also a fixed-point of $g^{-1}(x)$.

**b)** Let $r \in (a, b)$ be a fixed-point of $g(x)$. Show that if $|g'(r)| > 1$, then $|(g^{-1})'(r)| < 1$. Using the convergence theorem of fixed-point iteration, we now know that if the algorithm does not converge for $g(x)$, then it will converge for $g^{-1}(x)$ if one initialises with $x_0$ sufficiently close to $r$.

**c)** With this in mind, use fixed-point iteration to find an approximation to the solution $r$ of the equation $x = \arccos(x)$.

7 *Steffensen's method* for the solution of a non-linear equation $f(x) = 0$ with $f\colon \mathbb{R} \to \mathbb{R}$ is defined by the iteration

$$x_{k+1} = x_k - \frac{f(x_k)}{g(x_k)}$$

where

$$g(x_k) := \frac{f\big(x_k + f(x_k)\big) - f(x_k)}{f(x_k)}.$$

**a)** Write a Python implementation of Steffensen's method (use for instance the function `fixedpoint` or the function `newton` as a template). Test your implementation on the equation $f(x) = x - e^{-x}$ from problem 1.

**b)** Numerically estimate the convergence order of this method.