

## Practical Exercises 2: Nonlinear Equations

In the basic exercises, you will implement Newton's method and apply Newton's method as well as a pre-implemented bisection algorithm to two specific examples. Fast participants can additionally implement Newton's method with line search in Exercise 2.4.

The code is divided into a method's file `methods.py`, where the Newton and bisection algorithms are implemented, and a main file `main.py`, where these are applied to different examples. You find an initial version of both files in the download folder.

### Part I - Standard Exercises

#### Exercise 2.1

- (a) Implement the function `newton(f,df,x0,maxiter,tol)`, which computes a sequence of Newton iterates

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}, \quad k = 0, 1, \dots$$

to find a zero of a function  $f \in C^1([a, b])$  in the file `methods.py`. The function obtains the following parameters

- `f`: a function  $f : [a, b] \rightarrow \mathbb{R}$
- `df`: an (analytically given) derivative of  $f$
- `x0`: a starting value
- `maxiter`: maximum number of iterations
- `tol`: a tolerance for the stopping criterion

The iteration shall stop, when the stopping criterion  $|f(x_k)| < \text{tol}$  is fulfilled or when the maximum number of iterations is reached. In the latter case an error message shall be printed. The function `newton` returns the final iterate `x`, the number of iterations needed and a third return `fun_values`, which will remain unchanged for now and will be explained in part (c).

- (b) Test your implementation of `newton` for the function  $f(x) = \cos(x)$ , the starting value  $x_0 = \frac{5}{2}$ , `tol` =  $10^{-9}$  and `maxiter` = 1000, which need to be initialised in `main.py`. Uncomment the following code in 2.1(b), which prints your result `x`, the error  $|\mathbf{x} - \frac{\pi}{2}|$  and the number of iterations needed.

- (c) To analyse the convergence behaviour append the sequence of function values  $|f(x_k)|, k = 0, 1, \dots$  during the Newton iteration in `methods.py` to the array `fun_values`. Uncomment the lines in Section 2.1(c) in `main.py` that plot the array  $(|f(x_k)|)_{k=1}^n$  over  $k$  in a semi-logarithmic plot. Interpret the results by comparing the resulting graph with the results shown in the lecture.

## Exercise 2.2

- (a) The bisection method is implemented in the file `methods.py` as a function `bisection(f,a,b,tol)`, which takes the parameters

- `f`: a function  $f : [a, b] \rightarrow \mathbb{R}$
- `a, b`: the starting interval  $[a_0, b_0]$
- `tol`: a tolerance for the stopping criterion.

and returns the value

- the final iterate  $x_k$
- the number of iterations  $k$  needed
- the sequence of function values  $|f(x_k)|$  as an array

Uncomment the lines corresponding to Section 2.2(a) in `main.py` and initialise the parameters  $f(x) = \cos(x)$ ,  $a_0 = \frac{1}{2}$ ,  $b_0 = \frac{5}{2}$  and `tol` =  $10^{-9}$ . Run your program and compare your results with Newton's method.

- (b) Uncomment the lines in Section 2.2(b) of `main.py` that plot the function values  $|f(x_k)|$  together with the results from Newton's method obtained in 2.1(c) over the iteration  $k$  in a common plot. Interpret your results. Which convergence behaviour do you observe?

## Exercise 2.3

- (a) As a second example, we consider the function  $f_2(x) = \arctan(x)$ . Apply your implementation of the bisection method to this function by using the starting interval  $[a_0, b_0] = [-\frac{1}{2}, \frac{5}{2}]$  and `tol` =  $10^{-9}$ . Then use Newton's method with starting values  $x_0 = 2$ , `maxiter` = 1000 and the same tolerance. What do you observe? Interpret your results.
- (b) Try to find different starting values  $x_0 \neq 0$ , such that Newton's method converges.

## Part II - Extra exercises for fast participants

### Exercise 2.4

Implement Newton's algorithm with line search as presented in the lecture (Slide 30) in a new function

```
newton_linesearch(f,df,x0,q,maxiter,tol)
```

The additional parameter  $q \in (0, 1)$  is used for the criteria  $|f(x(\lambda))| < q|f(x_{k-1})|$  to define the step size  $\lambda$ . Apply your implementation to the function  $f_2$  from Exercise 2.3 with starting value  $x_0 = 2$  und  $q = 0.5$ . Print your step sizes  $\lambda$  and visualize the function values  $|f(x_k)|$  as in Exercise 2.1 (c). Interpret your results.