# MS-C1650 Numeerinen analyysi, Exercise 1, Guidelines

Lauri Perkkiö

April 26, 2019

- These are not model solutions, but "getting started guidelines".
- You are allowed, and supposed, to use MATLAB (or Octave/Python/whatever), unless otherwise stated. It is probably a good idea to bring a laptop with you to the exercise session.
- Ask questions at exercise sessions, or at certain office hours (TBA)
- Errors and stupidities (in these notes): contact lauri.perkkio@aalto.fi

## Exercise 1

**UPDATED Fri Apr 26. The converge plots for superlinear methods were wrong**

Find the largest root of $f(x) = x^3 - 5x + 3$, i.e., solve $f(x) = 0$ numerically. See Lecture notes (in Finnish) 3.1, 3.2, 3.3.

Try easy numbers ($x = -3, -2, ..., 3$), or sketch the graph of $f$, to get an initial guess for the root, $x_0$. For educational purposes, find the exact root $x_*$ by some means, so you can compare the error $e_k = |x_k - x_*|$ for each method and iteration.

**i) Bisection method.** (Puolitusmenetelmä). This is closest to a "trial-and-error-method". Find by trial $x_0$ and $x_1$ such that $f(x_0) < 0$ and $f(x_1) > 0$, and then continue bisecting (cutting in half) the interval to find the root. Record the error $e_k = |x_k - x_*|$ for each iteration.

**ii) Secant method.** (Sekanttimenetelmä). This is like Newton's iteration (see below), but the derivative of $f$ is approximated by a difference, instead of computed exactly:

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

You need two initial guesses, $x_0$ and $x_1$, which should optimally be chosen that the secant line through $f(x_0), f(x_1)$ intersects the root as close as possible. Again, record the error $e_k$.

**iii) Newton's method.** (Newtonin menetelmä). Like the above method, but the actual derivative is now available:

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

Again, record the error $e_k$.

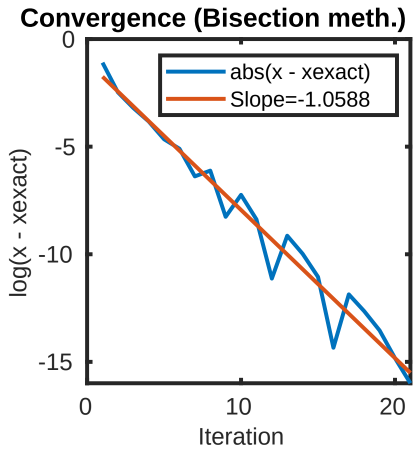
**Convergence rate, bisection method:** The convergence rate is expected to be linear, so roughly speaking

$$e_{k+1} \approx Ce_k \approx C^2 e_{k-1} \approx ... \approx C^k e_0.$$

Take logarithm:

$$\log e_{k+1} \approx \log e_0 + k \log C.$$

Thus, plotting $k$ vs $\log e_k$ should yield a line with the slope $\log C$ (which has no apparent meaning right now). My test run with this problem:



**UPDATED: Convergence rate, higher order methods:** The previous methods are expected to converge towards the root as

$$e_{k+1} \approx Ce_k^q,$$

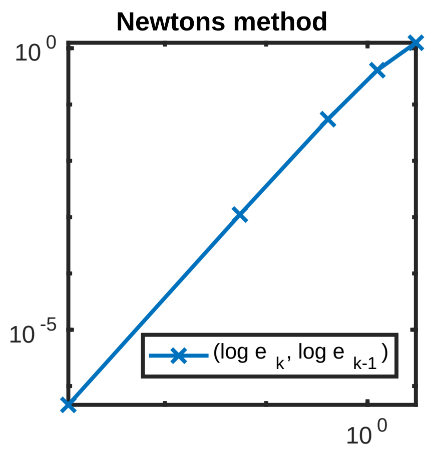
where $q$ is the convergence rate ($q = 1$ is linear, $q = 2$ is quadratic, secant method is somewhere in between). Take logarithm

$$\log e_{k+1} \approx \log C + q \log e_k,$$

and denote $y_k := \log e_k$. Then, compute that (direct computation, comes out by recursion)

$$(y_{k+1} - y_k)/(y_k - y_{k-1}) = q.$$

(Think of difference quotients; the derivative of something is a constant, $q$). This means that by plotting the points $(y_0, y_1), (y_1, y_2), ..., (y_n, y_{n-1})$ you should get a line with the slope $q$. My test run with Newton's method (the slope is 1.90, which is close to the theoretical $q = 2$):



## Exercise 2

Find the Lagrange polynomial passing through points $\{x_i, y_i\} = \{(1,2), (2,1), (3,6), (4,47)\}$. Lagrange polynomial is the (lowest degree) polynomial that passes through all the given data points. I.e., the polynomial $p(x)$ should be such that

$$p(x_i) = y_i \quad \text{for } i = 0, 1, 2, 3.$$

See Lecture notes 4.1. There are 4 data points, so the polynomial is (at most) degree $n = 3$.

**Lagrangian basis:** See the discussion at Definition 4.1.2 in the Lecture notes, and you can construct the interpolating polynomial in the Lagrangian basis. I.e., compute the polynomials

$$\varphi_i(x) = \prod_{i \neq j} \frac{x - x_j}{x_i - x_j},$$

and you immediately have the interpolating polynomial $p(x) = ...$

**Natural basis:** Could also be called the "monomial basis". Expand the previous polynomial into form $p(x) = c_3x^3 + c_2x^2 + c_1x + c_0$. In addition, use the "Vandermonde matrix"-style at Example 4.1.1 to compute the interpolation polynomial (or use whatever method that comes into your mind). You should obtain exactly the same polynomial. Remember: the lowest order polynomial passing through given data points is unique.

## Exercise 3

Idea: Given $R \neq 0$, we are computing

$$x = \frac{1}{R}, \tag{2}$$

but the division operation is not available. The iteration given in the assignment is

$$x_{n+1} = x_n(2 - x_nR).$$

What $f(x)$ is such that applying Newton's method $x_{n+1} = x_n - f(x_n)/f'(x_n)$ yields the previous formula? Naturally, $f(x) = 0$ should correspond to solving (2).

Then, proceed as in the assignment, and make a logarithmic plot as in Exercise 1 to see the quadratic convergence.

## Exercise 4

Compute

$$|e_2| \leq c|e_1||e_0| \leq ...$$

And continue

$$|e_n| \leq ... \to 0, \qquad \text{when } n \to 0,$$

remembering that $D < 1$. The residual $e_n$ tends to zero, i.e., the method converges.

## MATLAB

The assignment should be self-contained. Morale of the story: Some things are computed in practice by using "magic tricks", and this is one of them. This method to compute a cube root should converge extremely fast.

**Returning comprehensible solutions** that your fellow students are able to peer review: play around with "Publish"-button in MATLAB.

## CHALLENGE

"Challenge"-exercises are not discussed in the classroom, and they are not graded (as such). I.e., it is possible to get full points without returning these problems.

This problem involves four nonlinear equations with three unknowns $(x, y, z)$, i.e., the problem is overdetermined. One approach might be: how to solve overdetermined problems with Newton's method?