Introduction to Computational Physics

Matthias Jamin

(To be discussed on Friday, 16.5. and Monday, 19.5.)

EXERCISE 1.1: **Numerical accuracy**
Numerically, the derivative of a function $f(x)$ can be calculated with the difference quotient:

$$f'(x) \; = \; \frac{f(x+h)-f(x)}{h} + \mathcal{O}(h)\,,$$

or the improved symmetrical formula:

$$f'(x) \; = \; \frac{f(x+h)-f(x-h)}{2h} + \mathcal{O}(h^2)\,.$$

For h not very small, higher order corrections are still important, whereas for too small h, numerical uncertainties play a rôle, because one has to subtract two numbers which are almost equal. Calculate the derivative of the functions $\sin(x)$, $\exp(x)$ and $\ln(x)$ at $x=1$, and find the *optimal range* of n where $h \equiv 10^{-n}$ for the two cases in question.

EXERCISE 1.2: **Numerical stability**
As can be easily shown by partial integartion, the integrals

$$p_n \; = \; \int_0^1 x^n e^x \, dx$$

satisfy the recursion relation

$$p_{n+1} \; = \; e - (n+1)\, p_n \,,$$

with the boundary condition $p_1 = 1$. Using this recursion relation, calculate p_n up to $n = 20$. Compare with the exact result, and try to clarify the origin of the instability of the recursion relation. Employing the *inverse* recursion relation

$$p_n \; = \; (e - p_{n+1})/(n+1)\,,$$

with the artificial boundary condition $p_{30} = 1$, again calculate p_n for $n = 1 \ldots 20$. Compare with the previous results, and discuss your observation.

EXERCISE 1.3: **Functions**
Plot the following functions:

a) $f(x) \; = \; 3x^2 + \ln((\pi - x)^2)/\pi^4 + 1\,, \quad 0 < x < 2\pi$

b) $(x^2+y^2)\tan^{-1}(y/x) \; = \; 2y\,, \quad r \; = \; 2\sin(\theta)/\theta\,, \quad 0 < \theta < 4\pi$ (Spiral curve)

c) $x \; = \; 4\cos(\phi) + 2\cos(2\phi)\,, \quad y \; = \; 4\sin(\phi) - 2\sin(2\phi)\,, \quad 0 < \phi < 2\pi$ (Deltoid)

d) $(x^2+y^2)^2 \; = \; 9(x^2-y^2)\,, \quad r^2 \; = \; 9\cos(2\theta)\,, \quad 0 < \theta < 2\pi$ (Bernoulli's Lemniskate)

e) $r \; = \; \exp(\cos\theta) - 2\cos(4\theta) + \sin^5(\theta/12)\,, \quad 0 < \theta < 24\pi$ (Fey's butterfly curve)

EXERCISE 1.4: **Root Finding**
Implement the *Bisection Algorithm* for finding a root of a non-linear equation in a numerical programing language of your choice (e.g. C/C++, FORTRAN).