# Homework 2

## due January 31 2017, 23:59

**Task 1: Machine Epsilon**

In general a computer stores a real number in the following way

$$x = (-1)^s \cdot (0.a_1 a_2 ... a_t) \cdot \beta^e = (-1)^s \cdot m \cdot \beta^{e-t}, \quad a_1 \neq 0$$

where $s$ is either 0 or 1, $\beta$ (a positive integer larger than or equal to 2) is the *basis* adopted by the specific computer at hand, $m$ is an integer called *mantissa* whose length $t$ is the maximum number of digits $a_i$ (with $0 \leq a_i \leq \beta - 1$) that are stored, and $e$ is an integral number called the *exponent*. The numbers given in this form are called floating-point numbers, since the position of the decimal point is not fixed. The digits $a_1 a_2 ... a_p$ (with $p \leq t$) are called the $p$ first significant digits of $x$. The accuracy with which floating-point numbers are stored depends then on $\beta$ and $t$, and so does the amount of memory required to store them. For example, double precision real numbers are stored in registers of 8 Bytes: the sign $s$ is stored in 1 bit, the exponent $e$ in 11 bits, and the mantissa $m$ in 52 bits. Note that, although there are 52 bits for $m$, we can count $t = 53$ digits when $\beta = 2$. As a matter of fact, since the first digit $a_1$ of every floating point number must be different from 0, when $\beta = 2$ it is worthless to store it as it must necessarily be 1. A *round-off error* is inevitably generated whenever a real number $x \neq 0$ is replaced by its floating-point representative $x_{\text{num}}$, this error is always limited by

$$\frac{|x - x_{\text{num}}|}{|x|} \leq \frac{1}{2}\varepsilon,$$

where $\varepsilon = \beta^{1-t}$, called *machine epsilon*.

The following code can be used in MATLAB to compute $\varepsilon$.

```
numprec=double(1.0); % Define 1.0 with double precision
numprec=single(1.0); % Define 1.0 with single precision
while(1 < 1 + numprec)
    numprec=numprec*0.5;
end
numprec=numprec*2
```

a) Determine $\varepsilon$ using the above program, both for single and double precision.

b) Explain in detail what the code does. Why do we consider addition to 1?

c) Explain the difference between single and double precision. How many Bytes are used to store a single precision number? How many for the mantissa?

**Task 2: Round-off Error**

In this exercise, the errors involved in the numerical approximation of derivatives are examined. Using cental finite differences the derivative of a function $f(x)$ can be approximated as:

$$f'(x) \approx f'_{\text{num}}(x) = \frac{f(x+\Delta x) - f(x-\Delta x)}{2\Delta x} \tag{1}$$

a) Compute, numerically, the relative discretization error of the derivative of the function $f(x) = \dfrac{1}{1+x} + x$ using equation (1). The relative discretization error is given by:

$$\xi_{\text{d}} = \frac{|f'(x) - f'_{\text{num}}(x)|}{|f'(x)|}, \tag{2}$$

where $f'(x)$ is the analytical derivative of $f(x)$. Compute $\xi_d$ at $x = 2$ for different stepsizes $\Delta x \in [10^{-20}, 1]$. Use both single and double precision for the calculation and present the results in a double logarithmic plot[1] ($\xi_d$ vs. $\Delta x$). Remember that *all* variables used here should be defined as double or single precision as in Task 1.

b) The absolute propagation error $\xi_{\text{p}}$ of an arithmetic operation $\circ$ $(+, -, \times$ or $/)$ between two numbers $a_1$ and $a_2$ can be evaluated as: $a_1 \circ a_2 - a_{1,\text{num}} \circ a_{2,\text{num}}$, where $(\cdot)_{\text{num}}$ is the machine representation of the respective number.

Show that the propagation error of the addition of two positive numbers $a_1$ and $a_2$ is given by

$$\xi_{\text{p,add}} = \frac{a_1}{a_1 + a_2}\varepsilon_{a_1} + \frac{a_2}{a_1 + a_2}\varepsilon_{a_2}, \tag{3}$$

where $\varepsilon_{a_j} := (a_j - a_{j,\text{num}})/a_j$ is the machine accuracy on the quantity $a_{j,\text{num}}$.

A general formula for the propagation error for a function $g(a_1, a_2, \ldots, a_n)$ representing multiple arithmetic operations is given by:

$$\xi_{\text{p}} = \sum_{j=1}^{n} \left| \frac{a_j}{g} \frac{\partial g}{\partial a_j} \right| \varepsilon_{a_j}, \tag{4}$$

Show that when $g = a_1 + a_2$ this formula results in equation (3).

c) Show that, when using the proposed central differences approximation, the relative discretization error (equation (2)) is given by:

$$\xi_{\text{d}} \approx \frac{\Delta x^2 |f'''(x)|}{6|f'(x)|}$$

(Hint: Taylor expansion)
and that the propagation error (equation (4)) is given by:

$$\xi_{\text{p}} \approx \frac{|f(x)|\varepsilon}{|f'(x)|\Delta x}$$

where $\varepsilon$ is the machine accuracy. Find, analytically, the value of $\Delta x$ that minimizes the total error

$$\xi_{\text{tot}} = \xi_{\text{d}} + \xi_{\text{p}}.$$

Plot $\xi_{\text{d}}, \xi_{\text{p}}$ and $\xi_{\text{tot}}$ together with the results from part a).

[1] In MATLAB double logarithmic plots are obtained by the function `loglog()`.

**Task 3 : Discretization in time**

In this problem the stability and convergence order of three numerical time discretization methods is examined. Consider the first order, linear, test equation (the Dahlquist equation)

$$\begin{cases} u'(t) = f(u) = \lambda u(t), & 0 < t \leq T, \\ u(0) = 1 \end{cases} \tag{5}$$

where $\lambda = \lambda_{\Re} + i\lambda_{\Im} \in \mathbb{C}$. The time interval $[0, T]$ is discretized into $N$ equally spaced parts: $t_n = n\Delta t$, $n = 0, 1, \ldots, N$, where $\Delta t$ is the step-size. The following numerical methods should be used:

- explicit Euler
$$u^{n+1} - u^n = \Delta t f(u^n)$$
- implicit Euler
$$u^{n+1} - u^n = \Delta t f(u^{n+1})$$
- Crank-Nicolson
$$u^{n+1} - u^n = \frac{1}{2}\Delta t \left[f(u^{n+1}) + f(u^n)\right]$$

where $u^n := u(t_n)$.

a) Solve the system (5) analytically (by hand) to obtain the exact solution $u = u_{ex}$.

b) For $\lambda = -\sqrt{3}/2 + i\pi$ and for the five cases $N = 20, 40, 50, 100$, and $200$, compute the numerical solution iteratively until $T = 10$ for all the the three methods. Plot the real part of the solutions together with the exact solution for each value of $N$. What do you observe?

c) Now, consider $\lambda \in \mathbb{R}$. For each of the three considered schemes: (i) derive the expression of the amplification factor $G(z)$, where $z := \lambda\Delta t$; (ii) calculate $\lim_{z\to-\infty} G(z)$; (iii) plot $G(z)$ as a function of $z$ together with the result for the exact amplification over the interval $z \in [-10, 0.5]$. Discuss the performance of the schemes in the limits $z \to -\infty$ and $z \to 0$. Also, answer: why is the imaginary part of $\lambda$ irrelevant for this analysis?

d) For $\lambda = -\sqrt{3}/2 + i$, first do as in b) and explain the differences. Then, for each method, at a fixed time (chose $t = 3$) compute and plot the error $|u_{\text{ex}} - u_{\text{num}}|$ as a function of $N$ in a double logarithmic plot and estimate the order of accuracy by considering the slope of the curve. (Hint: $\log(x^p) = p\log(x)$.)

e) Based this task, discuss the usefulness, stability and accuracy of the methods.