Numerical Methods in Scientific Computing 2021

Exercise 1
Submit your solution to Moodle no later than Tuesday 26.1.2021 23:59
Exercise session: Thursday 28.1.2021

Problem 1. *(pencil and paper)* (6 points)
Assume that real numbers x and y have their machine presentations $\hat{x}$ and $\hat{y}$ in error by e_x and e_y:

$$\hat{x} \in [x - e_x, x + e_x] \quad , \quad \hat{y} \in [y - e_y, y + e_y] \quad .$$

The relative errors are defined as

$$r_x = \frac{x - \hat{x}}{x} = \frac{e_x}{x} \quad , \quad r_y = \frac{y - \hat{y}}{y} = \frac{e_y}{y} \quad .$$

How do the errors in x and y propagate when the numbers are multiplied or divided; i.e. calculate the relative errors of the product xy and quotient x/y in terms of r_x and r_y. You may assume $|r_x|, |r_y| \ll 1$.

Problem 2. *(computer)* (6 points)
Harmonic series is defined as

$$s = \sum_{k=1}^{\infty} \frac{1}{k} \quad .$$

We know that it diverges. However, when naively doing the summation with a computer starting from term $k=1$ and using single precision floating point numbers the value of the sum is finite (and surprisingly small).

A) Write a function named “`harmonic()`” that returns this sum, and calculate it. Remember to use single precision numbers (float in C, default real kind in Fortran)!
B) Write a modified version of your function “`harmonic_bunch(N)`” which adds the terms in bunches of N terms (you can use nested loops). What are the results if you do the summation for N=50, 100, or 500 terms (still starting from the term $k=1$)?

Submit the code of both of your functions in a file named “harmonic”

Problem 3. *(pencil and paper)* (6 points)
Show that the 2's complement really represents the additive inverse of a binary integer. Do it by proving that performing the operations

A) complement all bits
B) add one

on a binary integer produces a number that when added to the original one gives zero as a result.

Problem 4. *(computer/pencil and paper)* (6 points)

A) Write a function `funks(x)` that prints the value of the two following functions for a given double precision number *x*:

(a) $f(x)=\frac{\cos(x)-1}{x^2}$ (b) $f_2(x)=\frac{e^x-e^{-x}}{2x}$.

Using your code, examine how the computed values of t f_1 and f_2 behave near the point $x=0$. Name the source file “`funks`”.

B) In order to avoid loss of significance as a result of subtracting almost equal numbers, expand the functions in (a) into Taylor series and estimate the error of the approximation by using Taylor's theorem:

$f(x)=f(0)+\sum_{i=1}^{n-1}\frac{x^i}{i!}f^{(i)}(0)+R(n)$ where the error term is

$R(n)=\frac{x^n}{n!}f^{(n)}(\xi)$, $0\le\xi\le x$

C) Use the resulting approximation to calculate the values of the functions f_1 and f_2 near zero. For that purpose write a new function named `funks_Taylor(x)`. Include this function in the same source file as in Problem 1.