

Math 224: Scientific Computing

Homework 1: Float-point Numbers and Round-off Errors

Due: Thursday, September 4, 2008

1. Read Heath Chapter 1. Also, check out <http://www.nsc.liu.se/wg25/book/ch1/> on some dramatic pitfalls in scientific computing.
2. Write **C** or **Fortran** programs to determine ϵ_{mach} . Turn in your programs for (a), (b), and answer (c).
 - (a) First, for **float**- (in **C**) or **real**- (in **Fortran**) type variables: Declare two variables: **x**, **epsilon**. Initialize $\epsilon = 1.0$. Create a loop, each time through the loop, assign $x = 1.0 + \epsilon$ and print-out the values of $x - 1.0$ and ϵ . Then, each time through the loop, also halve the value of ϵ . Repeat until the value of $x - 1$ is no longer positive.
 - (b) Repeat for **double**-type variables (**double precision** if working in **Fortran**). If you are working in **Fortran**, remember to write the floating-point number 1.0, for example, as 1.0d0 to get double precision.
 - (c) Use the results of (a) and (b) to determine number of bits in the mantissa¹ of the two floating-point data types.
3. Write **C** or **Fortran** programs to determine the *overflow* level:
 - (a) First, the determine the largest possible **float**-type variables: Declare a **float** or **real** variable r and initialize $r = 1.0$. Create a loop, each time through the loop, print-out the values of r and $\log(r)/\log(2)$. (What is the meaning of the second quantity?) Then, each time through the loop, also double the value of r . Repeat until the value of r *overflows* the range of floats. In **C**, the function `isinf(x)`, which tests if a number has overflowed and become “infinite,” may be useful—it returns logical (integer) values 1 (TRUE) if “ $x = \infty$ ” else 0 (FALSE). I am not aware of an analogous **Fortran** subroutine on our system, although you can do `x .eq. x/0` to check if “ $x = \infty$.” Admittedly ad hoc, but it should work.

Find the value k for the largest 2^k before overflow.
 - (b) Repeat for **double**-type variables.
 - (c) Use the results of (a) and (b) to determine number of bits in the exponent of the two floating-point data types, and the overall total number of bits for each floating-point type number based on this question and on (2).
4. Use Taylor approximations to avoid the loss-of-significance error in the following computations for small x :

$$(a) \quad f(x) = \frac{e^x - e^{-x}}{2x}$$

$$(b) \quad f(x) = \frac{\ln(1-x) + xe^{x/2}}{x^3}$$

Box your final answers. In both cases, what is $\lim_{x \rightarrow 0} f(x)$?

¹The boxed items in homework questions are the final answer for each part—please box them on your problem sets to make things easier to grade.