Numerical Algorithms
Problem set 2: To err is human and to blame it on a computer is more like it.

August 29, 2024

[Note: In this tutorial we sometimes use $0.xyz\ldots$ as the normalization instead of $x.yz\ldots$ Sorry about the confusion!]

1 Chopping and Rounding

1. Consider a system of base 10 floating point numbers, with normalized 4-digits,with the exponent between -5 and 4. Determine the chopping and the rounding errors in representing the following numbers and try to relate the results to the stuff discussed in the class. (Note: the subscript C means that you are to chop 'em off, while R refers to the fact that you are to round 'em off.)

$$(765.4567)_C, (765.4567)_R, (765.499)_R, (100.05)_R$$

2. Consider the numbers

$$\begin{aligned} x_1 &= 0.1234 \times 10^1 \\ x_2 &= 0.3429 \times 10^0 \\ x_3 &= 0.1289 \times 10^{-1} \\ x_4 &= 0.9895 \times 10^{-3} \\ x_5 &= 0.9763 \times 10^{-5} \end{aligned}$$

   Add these numbers using four-decimal-digit chopped floating point arithmetic in both forward and reverse. Which is more accurate? Why?

3. Suggest methods for evaluating each of

   (a) $e^x \simeq 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots$

(b) $\cos x \simeq 1 - \frac{x^2}{2!} + \frac{x^4}{4!} + \ldots$

(c) $\sin x \simeq x - \frac{x^3}{3!} + \frac{x^5}{5!} + \ldots$

for $x = 25$ up to 12 digits of accuracy. Pay attention to both truncation and round-off errors. Try out by writing programs.

4. Suppose you have to evaluate

$$F(x) = x \sin x/(1 - \cos x)$$

near $x = 0$. Do you foresee any problems? Suggest a method to overcome the problem. Again try out by programming.

2 Computer Arithmetic

Computing $10.1 - 9.93$ with $p = 3$ we get,

$$\begin{aligned} x &= 1.01 \times 10^1 \\ y &= 0.99 \times 10^1 \\ x - y &= 0.02 \times 10^1 \end{aligned}$$

The correct answer is 0.17, so the computed difference is wrong in every digit! How bad can the error be?

Suppose that one extra digit is added to guard against this situation (a guard digit). That is, the smaller number is truncated to $p + 1$ digits, and then the result of the subtraction is rounded to p digits. With a guard digit, the previous example becomes

$$\begin{aligned} x &= 1.010 \times 10^1 \\ y &= 0.993 \times 10^1 \\ x - y &= 0.17 \times 10^0 \end{aligned}$$

which is better.

1. Prove that *Using a floating-point format with parameters β and p, and computing differences using p digits, the relative error of the result can be as large as $\beta - 1$.*

2. Prove that *If x and y are floating-point numbers with parameters β and p, and if subtraction is done with $p + 1$ digits (i.e. one guard digit), then the relative rounding error in the result is less than 2ϵ.*

3. Evaluate and estimate the errors in the following cases. Assume the base to be 10, with normalized 4-digits and the provision of a guard-bit. (In the case of differnce compare your answer with the case where you have (costly!) registers with 4 guard-bits, and also the case where you have no guard bits).

$$(0.1004 \times 10^0) - (0.9958 \times 10^{-1}), (0.1234 \times 10^0) \times (0.1000 \times 10^1)$$

4. Prove that the presence of a guard digit ensures that a computed sum differs from the correctly chopped value by no more than unity in the p^{th} digit.

5. Let * denote any of the operations +,-,×,÷.Let $fl(x * y)$ represent the computed value of $(x * y)$. Assuming that x and y represent **normalized floating point numbers** and $y \neq 0$ when we consider division, show

$$\frac{|(x * y) - fl(x * y)|}{|x * y|} \leq r\mu$$

where,
$r = 1$ in ideal rounded arithmetic
$r = 2$ in ideal chopped arithmetic or during multiplication and division with a guard digit
$r = 4$ during addition or subtraction with a guard digit, or single-length register multiplication and division.
$r \geq 4$ in single-length register addition and subtractions.

6. Show that the unit rounding error μ can also be defined to be the largest number so that $fl(x + 1.0) = 1.0$ for every x with $0 \leq x < \mu$. Write a computer program that will determine the unit rounding error by using this definition. Take care that binary-to-decimal conversion does not affect your result.

3 Accumulation of Errors

1. Do a backward and forward error analysis on the computation of a product of five non-zero numbers: $p = x_1x_2x_3x_4x_5$

2. **Theorem** Let $\epsilon_1, \epsilon_2, ..., \epsilon_n$ be rounding errors that satisfy $|\epsilon_k| \leq r\mu$, $k = 1, 2, \ldots, n$. If $nr\mu \leq 0.1$, then there is a δ_n that satisfies the following conditions

$$\frac{(1+\epsilon_1)...(1+\epsilon_k)}{(1+\epsilon_{k+1})...(1+\epsilon_n)} = 1+\delta_n$$

and

$$\delta_n \leq n(1.06r\mu)$$

Try proving the above theorem by proving these lemmas:

Lemma 1 If $0 \leq r\mu \leq 1$, then $1 - nr\mu \leq (1-r\mu)^n$.
{*Hint:* Expand $f(y) = (1-y)^n$ in a Taylor's series}

Lemma 2 If $0 \leq x \leq 0.1$, then $1 + x \leq e^x \leq 1 + 1.06x$.
{*Hint:* Expand e^x in a Taylor's series.}

Lemma 3 If $0 \leq nr\mu \leq 0.1$, then $(1+r\mu)^n \leq 1 + 1.06nr\mu$.
{*Hint:* Use Lemma 2}

Lemma 4 If $|\epsilon_i| \leq r\mu$ for i=1,2,....,n, and $nr\mu 0.1$, then
$1 - nr\mu \leq (1+\epsilon_1)(1+\epsilon_2)....(1+\epsilon_n) \leq 1 + 1.06nr\mu$.

Lemma 5 If $0 \leq x \leq 0.1$, then $(1-x)^{-1} \leq 1 + 1.06x$.

Lemma 6 If $|\epsilon_i| \leq r\mu$ for i=1,2,....,n and $nr\mu \leq 0.1$, then

$$1 - nr\mu \leq \frac{1}{(1+\epsilon_1)...(1+\epsilon_n)} \leq 1 + 1.06nr\mu$$

{*Hint:* Use Lemmas 4 and 5}

Now combine Lemmas 4 and 6 to prove the above Theorem.