Computer Arithmetic and errors

MTHBD/CMPBD 423

- Truncation Error

$$e^x \approx 1 + x + \frac{x^2}{2} + \frac{x^3}{6}$$

- Round-Off Error

  When the true value of a number is not stored exactly by a computers representation, (either by conversion to a nonterminating fraction, in the computers base, or input beyond precision) the error associated with this imperfection is called **Round-Off Error**.

- Error in Original Data

  In mathematical models, you may have coefficients that are obtained by imperfect *real world* measurements. Or perhaps the model does not perfectly reflect the real world. Nothing we can do about this but is is worth doing a **sensitivity analysis** on the parameters.

- Propagated Errors:

  Suppose a real number x is represented by the machine as x^* where $x^* = x(1+\delta)$ and δ (the initial relative error) is small.

  Suppose I want to calculate x^2. In the machine I'll get $(x^*)^2 = x^2(1+\delta)^2$. Now I get an error $E = (x^*)^2 - x^2 = x^2\left[(1+\delta)^2 - 1\right] \approx x^2(2\delta)$ which can be very big, especially if x is big. If I look at relative error = $\frac{E}{x^2}$, I still get a relative error of 2δ. Notice, the relative error doubled. This is an example of an error being propagated.

- Computer Arithmetic
  - machine numbers: Three parts: the *sign*, the fraction part called the *mantissa or significand*, and the exponent called the *characteristic*.

$$\pm .d_1 d_2 d_3 \ldots d_p * B^e$$

    * $B =$ the base that is used generally: 2, 16, 10
    * d_i's digits or bits $0 < d_1 < B$ and $0 \leq d_i < B$ for $i \neq 1$.
    * $p =$ *the precision* the number of significand bits.
    * $e =$ the *characteristic* maybe 7 bits long.
  - MATLAB's max and min values (default is double precision):
    * The maximum positive number is $\approx 10^{308}$.
    * The smallest positive number is $\approx 10^{-100}$.
    * The machine eps is $\approx 10^{-16}$. Machine eps is the largest number such that: 1 + machine eps = 1.
  - errors in conversion (round-off error)
    In base 2, $\frac{1}{10}$ is the repeating decimal $.\overline{1100}_2$. It will never be stored exactly in a base 2 machine.
    In MATLAB: $|10{,}000 - \sum_{k=1}^{100{,}000} \frac{1}{10}| \approx 2 \cdot 10^{-8}$
  - relative and absolute error
    If p^* is the machine's approximation to p:
    * **absolute error** = $|p - p^*|$
    * **relative error** = $\frac{|p-p^*|}{|p|}$ provide $p \neq 0$.

– Arithmetic Accuracy (loss of significance)

Suppose we are using base 10, five digit (chopping) arithmetic.

variable	Actual Value	computer Value
x	1/3	0.33333 x 10^0
y	5/7	0.71428 x 10^0
u	0.714251	0.71425 x 10^0
v	98765.9	0.98765 x 10^5
w	0.111111 x 10^{-4}	0.111111 x 10^{-4}

computer operation	cumputer Result	Actual Value	Absolute Error	Relative Error
$x+y$	0.10476 x 10^1	22/21	0.190 x 10^{-4}	0.182 x 10^{-4}
$y-x$	0.38095 x 10^0	8/21	0.238 x 10^{-5}	0.625 x 10^{-5}
$x\cdot y$	0.23809 x 10^0	5/21	0.524 x 10^{-5}	0.220 x 10^{-4}
$y\div$ x	0.21428 x 10^1	15/7	0.571 x 10^{-4}	0.267 x 10^{-4}
$y-u$	0.30000 x 10^{-4}	0.34714 x 10^{-4}	0.471 x 10^{-5}	0.136
$(y-u)\div w$	0.27000 x 10^1	0.31243 x 10^1	0.424	0.136
$(y-u)\cdot v$	0.29629 x 10^1	0.34825 x 10^1	0.465	0.136
$u+v$	0.98765 x 10^5	0.98766 x 10^5	0.161 x 10^1	0.163 x 10^{-4}

lesson: Subtracting similar values creates a large relative error. This is called a **loss of significance**. Also, adding two numbers of disparate size causes large absolute error. For example in MATLAB: $1,000,000+10^{-11}=1,000,000$.

– Avoiding loss of significance due to subtracting similar numbers in the quadratic formula.

Recall the roots of ax^2+bx+c can be found by

$$x_1=\frac{-b+\sqrt{b^2-4ac}}{2a} \quad \text{and} \quad x_2=\frac{-b-\sqrt{b^2-4ac}}{2a} \tag{1}$$

Suppose $b^2-4ac>0$, $b>0$ and $b\approx\sqrt{b^2-4ac}$. There is thus a good chance of x_1 containing a large error, especially if a is relatively small. We resolve this by changing the form of x_1 by *rationalizing the numerator*

$$x_1=\frac{-b+\sqrt{b^2-4ac}}{2a}\left(\frac{-b-\sqrt{b^2-4ac}}{-b-\sqrt{b^2-4ac}}\right)=\frac{-2c}{b+\sqrt{b^2-4ac}} \tag{2}$$

With the above form for x_1 the subtraction of nearly equal numbers is eliminated. Therefore, it would be wise to calculate x_1 by equation (2) and x_2 by equation (1). A similar revision is needed if $b<0$ in which case, x_2 from equation (1) is at risk of losing precision due to subtracting similar numbers.

– Avoiding round off error with nested multiplication

  Even in the absence of subtracting similar numbers, there will be round-off error and the best way to minimize this is to reduce the number of computer operations. A good example of this is nested multiplication.

  Consider the function $f(x) = x^3 - 6x^2 + 3x - 0.149$ evaluated at $x = 4.71$. We will assume three digit base 10 arithmetic for the machine computations.

  * traditional evaluation: 5 multiplies and 3 adds/subtracts.
    The relative error with either chopping or rounding is ≈ 0.04
  * nested multiplication: 2 multiplies and 3 adds/subtracts.

$$f(x) = ((x-6)x+3)x - 0.149 \tag{3}$$

    The relative error with chopping is ≈ 0.0093
    The relative error with rounding is ≈ 0.0025

- Error analysis of algorithms generally assumes perfect precision, ie. no round-off error. However, it is there and is worth keeping it in mind. Especially if you are doing many sequential calculations where the output from one is input into another. In this way, errors can be propagated and your final answer can be garbage.