Lecture 1: Errors in Numerical Computation

Department of Mathematics
IIT Guwahati

Rajen Kumar Sinha

Real Numbers, Machine Numbers, and Rounding

- True values are sometimes not stored exactly by a computers representation. This is called round-off error. The actual number that is stored in the computer may undergo chopping or rounding of the last digit.
- Additional errors are introduced when the arithmetic operations are carried out on the computer.

Real Numbers: In general, a nonzero real number x can be represented in the form (normalized scientific notation)

$$x = \pm r \times 10^n,$$

where r is a number in the range $\frac{1}{10} \leq r < 1$ and $n \in \mathbb{Z}$. Of course, if $x = 0$, then $r = 0$; in all other cases, we can adjust n so that r lies in the given range.

Example

- $123.4567 = .1234567 \times 10^3$
- $-0.0004321 = -.4321 \times 10^{-3}$

In the binary system.

$$x \in \mathbb{R} \iff$$
$$x = \pm \left(b_n 2^n + b_{n-1} 2^{n-1} + \cdots + b_0 + b_{-1} 2^{-1} + b_{-2} 2^{-2} + \cdots \right),$$

where $n \geq 0$ is some interger and the binary digits b_i takes the value either 0 or 1,

$$b_i = 0 \text{ or } b_i = 1 \text{ for all } i.$$

In general,

$$x = \pm (b_n b_{n-1} \cdots b_0 \cdot b_{-1} b_{-2} b_{-3} \cdots)_2,$$

where 2 reminds us that we are dealing with a binary number.

Example

- $(1011.01)_2 = 2^3 + 2^1 + 2^0 + 2^{-2} = 8 + 2 + 1 + 1/4 = (11.25)_{10}$
- $(.010\overline{101}\ldots)_2 = \sum_{\substack{k=2 \\ (k=\text{even})}}^{\infty} 2^{-k} = \sum_{m=1}^{\infty} 2^{-2m} = \frac{1}{4} \sum_{m=0}^{\infty} (\frac{1}{4})^m$
$$= \tfrac{1}{4} \cdot \tfrac{1}{1-\frac{1}{4}} = \tfrac{1}{3} = (0.33\bar{3} \cdots)_{10}$$
- $\frac{1}{5} = (0.2)_{10} = (0.00110\overline{0011}\ldots)_2$

Machine Numbers: There are two kinds of machine numbers.

- Floating-point number
- Fixed-point number

Floating-point number:

$\mathbb{R}(t, s)$ – The set of (real) floating-point numbers on a computer.

$$x \in \mathbb{R}(t, s) \iff x = f \cdot 2^e, \quad \text{where}$$

$$f = \pm(\cdot b_{-1} b_{-2} \cdots b_{-t})_2, \quad e = \pm(c_{s-1} c_{s-2} \cdots c_0)_2,$$

where all b_i and c_j are binary digits, that is, either zero or one. In the above, t be the number of binary digits allowed by the computer in the fractional part and s be the number of binary digits in the exponent. f is referred to as the mantisha and e (the integer) as the exponent of x.

A floating-point number is accommodated in a machine register is shown below.

$\pm$	b_{-1}	b_{-2}	$\cdots$	b_{-t}	$\pm$	c_{s-1}	c_{s-2}	$\cdots$	c_0

The largest and smallest magnitute of a floating-point number is given by

$$\begin{aligned} fl_{max}(x) &:= \max_{x\in\mathbb{R}(t,s)} |x| = (1-2^{-t})\,2^{2^s-1}, \\ fl_{min}(x) &:= \min_{x\in\mathbb{R}(t,s)} |x| = 2^{-2^s}. \end{aligned} \tag{1}$$

- A nonzero real number whose modulus is not in the range of (1) cannot be represented on this particular computer.
- During the course of computation, if a number produced whose modulus is larger than $fl_{max}(x)$ then we say that overflow has occurred. If its molulus is smaller than $fl_{min}(x)$ then we say that underflow has occurred.

- To increase the precision, one can use two machine registers to represent a machine number, and call $x \in \mathbb{R}(2t, s)$ a double-precision number.

Fixed-point numbers: This is the case $x = f$, i.e., $e = 0$.

Rounding: Let

$$x \in \mathbb{R}, \quad x = \pm \left(\sum_{k=1}^{\infty} b_{-k} 2^{-k} \right) 2^{e} \tag{2}$$

be the exact real number (in normalized floating-point form) and

$$x \in \mathbb{R}(t, s), \quad x^* = \pm \left(\sum_{k=1}^{t} b^*_{-k} 2^{-k} \right) 2^{e^*} \tag{3}$$

be the rounded number.

Chopping: $x^* = \text{chop}(x)$, $e^* = e$, $b^*_{-k} = b_{-k}$, $k = 1, 2, \ldots, t$.

Symmetric rounding: The rounding up or rounding down in decimal arithmetic based on the first discarded decimal digits. If the discarded digit ≥ 5, one rounds up; if it is < 5 , one rounds down.

Example. If two decimal-digit floating-points number are used, then

$$fl(2/3) = \begin{cases} (0.67) \cdot 10^0, & \text{rounded} \\ (0.66) \cdot 10^0, & \text{chopped} \end{cases}$$

$$fl(-249) = \begin{cases} -(0.25) \cdot 10^3, & \text{rounded} \\ -(0.24) \cdot 10^3, & \text{chopped} \end{cases}$$

Example. The real number

$$x = \frac{22}{7} = 3.142857142857142857\cdots$$

has the following six-digit representations:

$$fl(x) = \begin{cases} 0.314286 \times 10^1, & \text{rounded}, \\ 0.314285 \times 10^1, & \text{chopped}. \end{cases}$$

Note. In binary arithmetic, if the first discarded binary digit is 1, in which case one rounds up; if it is 0, then one rounds down.

In terms of the chop operation,

$$x^* = fl_{chop}(x), \quad fl_{chop}(x) := chop\left(x + \frac{1}{2} \cdot 2^{-t} \cdot 2^e\right).$$

In the case of chopping, the absolute error is

$$\begin{aligned} |x - fl_{chop}(x)| &= \left| \pm \sum_{k=t+1}^{\infty} b_{-k} 2^{-k} \right| 2^e \\ &\leq \sum_{k=t+1} 2^{-k} \cdot 2^e = 2^{-t} \cdot 2^e. \end{aligned} \tag{4}$$

The relative error can be estimated as

$$\left| \frac{x - fl_{chop}(x)}{x} \right| \leq \frac{2^{-t} \cdot 2^e}{|\pm \sum_{k=1}^{\infty} b_{-k} 2^{-k}| \, 2^e} \leq \frac{2^{-t} \cdot 2^e}{\frac{1}{2} \cdot 2^e} \leq 2 \cdot 2^{-t}.$$

Similarly, in the case of rounding, one finds

$$\left| \frac{x - fl_{round}(x)}{x} \right| \leq 2^{-t}. \tag{5}$$

Remark. Note that the absolute error (4) depends on e (the magnitude of x), which is the reason why one prefers the relative error. The number on the right is an important, machine-dependent quantity, called the machine precision ($\delta = \frac{p}{2}2^{1-t}$, $p = 1$ for rounding and $p = 2$ for chopping).

Set $\epsilon = \frac{fl(x)-x}{x}$. Then

$$fl(x) = x(1 + \epsilon), \quad |\epsilon| \leq \delta.$$

Machine Arithmetic. Let $\circ$ denote arithmetic operation $(=, +, -, \times, /)$. Let $x, y \in \mathbb{R}(t, s)$ and let $fl(x \circ y)$ denote the machine produced result of the arithmetic operation. Then

$$fl(x \circ y) = x \circ y(1 + \epsilon), \quad |\epsilon| \leq \delta. \tag{6}$$

Multiplication: Consider values $x(1+\epsilon_x)$ and $y(1+\epsilon_y)$ of x and y contaminated by relative errors ϵ_x and ϵ_y, respectively.

Q. What is the relative error in the product?

$$\begin{aligned} x(1+\epsilon_x)\cdot y(1+\epsilon_y) &= x\cdot y(1+\epsilon_x+\epsilon_y+\epsilon_x\epsilon_y) \\ &\approx x\cdot y(1+\epsilon_x+\epsilon_y). \end{aligned}$$

Assuming ϵ_x and ϵ_y are sufficiently small, neglecting second $(\epsilon_x^2, \epsilon_x\epsilon_y, \epsilon_y^2)$, and higher order term, the realtive error $\epsilon_{x\cdot y}$ is given by

$$\epsilon_{x\cdot y} = \epsilon_x + \epsilon_y.$$

Thus, we see that the (relative) errors in the data are being added to produce the (relative) error in the result. We consider this to be acceptable error propagation, and in this sense, multiplication is a benign (acceptable) operation.

Division:

$$\begin{aligned}\frac{x(1+\epsilon_x)}{y(1+\epsilon_y)} &= \frac{x}{y}(1+\epsilon_x)(1-\epsilon_y+\epsilon_y^2-+\cdots) \\ &\approx \frac{x}{y}(1+\epsilon_x-\epsilon_y),\end{aligned}$$

In this case, $\epsilon_{x/y} = \epsilon_x - \epsilon_y \Longrightarrow$ division is also a benign operation.

Addition and Subtraction:

$$\begin{aligned}x(1+\epsilon_x)+y(1+\epsilon_y) &= x+y+x\epsilon_x+y\epsilon_y \\ &= (x+y)(1+\frac{x\epsilon_x+y\epsilon_y}{x+y}),\ \ x+y\neq 0.\end{aligned}$$

$$\epsilon_{x+y} = \frac{x}{x+y}\epsilon_x + \frac{y}{x+y}\epsilon_y. \tag{7}$$

Note that the coefficients of ϵ_x and ϵ_y can assume large values. If x and y have same sign, $|\frac{x}{x+y}|, |\frac{y}{x+y}| < 1$, then

$$|\epsilon_{x+y}| < |\epsilon_x| + |\epsilon_y|.$$

In this case, addition is a benign operation. When x and y have opposite sign, the ratio $\frac{x}{x+y}, \frac{y}{x+y}$ can be arbitrarily large when $|x+y| << |x|$ and $|y|$. The large magnification error in (7) is referred as cancellation error.

Example 1. Consider the algebraic identity: $(a-b)^2 = a^2 - 2ab + b^2$. On 2-decimal-digit computer, with $a = 1.8$ and $b = 1.7$, we note that

$$fl(a^2 - 2ab + b^2) = 3.2 - 6.2 + 2.9 = -0.10, \quad (a-b)^2 = 0.010.$$

Example 2. $y = \sqrt{x+\delta} - \sqrt{x}$, $x > 0$, $|\delta|$ is very small. To avoid cancellation error, one should write

$$y = \frac{\delta}{\sqrt{x+\delta} + \sqrt{x}}.$$

Example 3. If $y = \cos(x+\delta) - \cos(x)$, then one should write

$$y = -2 \sin\frac{\delta}{2} \sin(x + \frac{\delta}{2}).$$

Loss of Significance

Consider the two numbers

$$x = 3.1415926536 \text{ and } y = 3.1415957341,$$

which are nearly equal and both carry 11 decimal digits of precision. Suppose that their difference is formed:

$$x - y = -0.0000030805.$$

Since the first six digits of x and y are the same, their difference $x - y$ contains only five decimal digits of precision. This phenomenon is called loss of significance.

Example. Let's compare the results of calculating $f(500)$ and $g(500)$ using six digits and rounding. The functions are

$$f(x) = x(\sqrt{x+1} - \sqrt{x}) \text{ and } g(x) = \frac{x}{\sqrt{x+1} + \sqrt{x}}.$$

For the first function,

$$\begin{aligned} f(500) &= 500(\sqrt{501} - \sqrt{500}) \\ &= 500(22.3830 - 22.3607) \\ &= 500(0.0223) = 11.1500. \end{aligned}$$

For $g(x)$,

$$g(500) = \frac{500}{\sqrt{501} + \sqrt{500}} = \frac{500}{44.7437} = 11.1748.$$

The second function, $g(x)$, is algebraically equivalent to $f(x)$.
The answer, $g(500) = 11.1748$, involves less error and is the same as that obtained by rounding the true answer $11.174755300747198\cdots$ to six digits.

End