Roundoff Error

On a computer, we can only represent #'s with a finite number of digits. This introduces a source of error into our calculations.

Digital computers use binary, e.g. for $x \in \mathbb{R}$,

$$x = \pm (q_n 2^n + q_{n-1} 2^{n-1} + \dots + q_0 2^0 + q_{-1} 2^{-1} + \dots)$$

for some $\{q_i\}$, $n \in \mathbb{Z}$.

EX: $18.5 = 1\times 2^4 + 0\times 2^3 + 0\times 2^2 + 1\times 2^1 + 0\times 2^0 + 1\times 2^{-1}$

$= (10010.1)_2$ Δ

We will use decimal to illustrate the main points, understanding that computers use binary for technical reasons.

Def: The **word length** of a finite precision representation is the # of digits n used to represent a number $x \in \mathbb{R}$.

In general, the word length is machine-dependent.

Def: **Fixed-point** representation specifies a fixed # n_1 of digits before the decimal point and a fixed # n_2 after the decimal point ($n_1 + n_2 = n$).

EX: $n = 10$, $n_1 = 4$, $n_2 = 6$

$30.421 = \quad 0030.421000$

$0.0437 = \quad 0000.043700$ Δ

(n_1 | n_2)

Clearly lots of wasted 0's!

Modern computers instead use floating point representations.

Def: In **floating point representation**, the decimal point position is specified with respect to the first digit of $x \in \mathbb{R}$. This is accomplished with use of an **exponent**,

$$x = a \times 10^b \text{ (or } 2^b\text{) with } |a| < 1,\ b \in \mathbb{Z}.$$

b is called the **exponent**, and a the **mantissa**

Ex: $30.421 = 0.30421 \times 10^2$ △

In floating point representation, there is a finite (integer) # of digits t for the mantissa and e for the exponent so that $n = t+e$ with n the word length.

Ex: $t=4$, $e=2$ for $x = 5420$

$$x = \underbrace{0.5420}_{a} \times 10^{\underbrace{04}_{e}} = \boxed{5420}_{10}\ \boxed{04}$$

Note: Floating point representation Not Unique!

$$= \boxed{0542}_{10}\ \boxed{05}$$ △

Def: We say a floating point representation is **normalized** if the first digit (bit) of the mantissa is nonzero. Then $|a| \geq 10^{-1}$ (2^{-1}). The **significant digits** (bits) of a number are the digits of the mantissa not counting leading zeros.

From now on, we only consider normalized arithmetic.

Def: For a fixed # of places t for the mantissa, e for the exponent, and a base B (e.g. $B=10$ or $B=2$), we call the set $A(t, e, B) \subsetneq \mathbb{R}$ of numbers exactly representable the machine numbers.

Note: A is finite!

Q: How do we approximate $x \notin A$ by $y \in A$?

Note: For $x, y \in A$, there are examples for which

$$x+y \notin A, \quad x/y \notin A, \quad x \cdot y \notin A, \quad x-y \notin A.$$

Def: Let $rd(x)$ denote a projection of $x \in \mathbb{R}$ onto A. We define this operation by

$$|x - rd(x)| \leq |x - y| \text{ for all } y \in A.$$

We generically obtain $rd(x)$ by rounding.

Ex: $t = 4$

$$rd(0.14285 \times 10^{0}) = 0.1429 \times 10^{0}$$
$$rd(3.14159 \times 10^{0}) = 0.3142 \times 10^{1}$$
$$rd(0.142842 \times 10^{2}) = 0.1428 \times 10^{2} \quad \text{A}$$

To round, write $x \in \mathbb{R}$ normalized as $x = a \cdot 10^{b}$ with $|a| < 10^{-1}$.

If $|a| = 0.\gamma_1\gamma_2 \cdots \gamma_i \gamma_{i+1} \cdots$, $0 \leq \gamma_i \leq 9$, $\gamma_1 \neq 0$

we form

$$a' = \begin{cases} 0.\gamma_1\gamma_2 \cdots \gamma_t & \text{if } 0 \leq \gamma_{t+1} \leq 4 \\ 0.\gamma_1\gamma_2 \cdots \gamma_t + 10^{-t} & \text{else} \end{cases}$$

and set $\tilde{rd}(x) = \text{sgn}(x) \cdot a' \times 10^{b}$.

Note:

$$\frac{|\widetilde{rd}(x) - x|}{|x|} = \frac{|\mathrm{sgn}(x)\, a' \cdot 10^{b} - \mathrm{sgn}(x)\, a \cdot 10^{b}|}{|a| \cdot 10^{b}}$$

$$= \left| \frac{a' - a}{a} \right|$$

$$\leq \frac{5 \cdot 10^{-(t+1)}}{|a|}$$

$$\leq \underbrace{5 \cdot 10^{-t}}_{\hat{=} \varepsilon} \qquad (|a| \geq 10^{-1})$$

We call this number ε or **machine epsilon**.
On modern CPU's, $\varepsilon \approx 10^{-16}$. On GPU's, 10^{-8}.
We can then always write that

$$\widetilde{rd}(x) = x(1+\delta) \quad \text{with} \quad |\delta| \leq \varepsilon.$$

If $\widetilde{rd}(x) \in A$, we define $rd(x) = \widetilde{rd}(x)$.

Note: Some $x \in \mathbb{R}$ are too large or too small, so that

$$\widetilde{rd}(x) \notin A$$

this is called **overflow** and **underflow**.

Ex: $t = 4$, $e = 2$

$$\widetilde{rd}(0.31794 \cdot 10^{110}) = 0.3179 \times 10^{110} \notin A.$$

Def: We define the floating point operations $\oplus, \ominus, \otimes, \oslash$ $: A \times A \to A$ by

$$x \oplus y = rd(x+y)$$

$$x \ominus y = rd(x-y)$$

$x \otimes y = rd(x \cdot y)$

$x \oslash y = rd(x / y)$

Then, $x \oplus y = (x + y)(1+\delta)$, $|\delta| \leq \varepsilon$ with analogous claims for the other operations.

Q: What is $1 \oplus 10^{-16}$?

A: 1 (try in Python!)

Q: Is $\oplus$ associative?

A: No! $(0.1 \oplus 10^{-16}) \oplus 1 \neq 0.1 \oplus (10^{-16} \oplus 1)$.