Lecture # 4

01/18/2023

Not really Algorithms : errors in computer :

How to quantify errors?

Two ways: $\rightarrow$ Absolute error ✓

$\rightarrow$ Relative error

$x_{true} = \pi$ $\quad\quad$ $x_{approx} = 3.141592$

$$\text{Absolute error} = |x_{true} - x_{approx}|$$

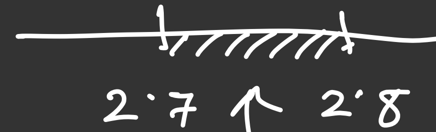
$$\text{Relative error} := \frac{|x_{true} - x_{approx}|}{|x_{true}|}$$

$$= \frac{\text{Absolute error}}{|x_{true}|} \in [0, 1]$$

e.g., 0.07 ← can be interpreted as % error (e.g., 7% error)

∴ Relative error × 100 can be interpreted as the % error.

Real numbers:



Uncountably many real numbers between any two given real numbers

But in digital computer, only finite precision can be stored/represented

Decimal numbers $\longleftrightarrow$ Binary numbers

$(9)_{10} \longleftrightarrow (1001)_2$

$= 9 \times 10^0 \longleftrightarrow 9 = (1 \times 2^3) + (0 \times 2^2) + (0 \times 2^1) + (1 \times 2^0)$

Fractions:

$(0.375)_{10} = (?)_2$

$0.375 \times 2 = 0.\boxed{750}$ → 0

$0.750 \times 2 = 1.\boxed{500}$ → 1

$0.500 \times 2 = 1.000$ → 1

$\therefore (0.375)_{10} = (0.\boxed{011})_2$

$\therefore \quad (9.375)_{10} \longleftrightarrow (1001.011)_2$

Need to standardize how binary representations of real numbers should be stored in a computer:

IEEE 754 floating point format:

(standardization)

Precision	Sign	Mantissa	Exponent	
Single	1 bit	23 bits	8 bits	$\rightarrow$ total 32 bits
Double	1 bit	52 bits	11 bits	$\rightarrow$ total 64 bits

By default, MATLAB uses double precision:

Storing binary numbers:

$$x = \underset{\text{Sign}}{\pm}\ 1.\underbrace{bbb\ldots b}_{\text{Mantissa}} \times 2^{\overset{\text{exponent}}{p}}$$

Example:

$$(9)_{10} = (1001)_2$$

$$= +1.001 \times 2$$

Round off error is inevitable:

Not to be confused with undefined / illegal math operations:

>> Inf − Inf = NaN ← Not a number

>> $\dfrac{\text{Nonzero real}}{0} = \pm\,\text{Inf}$

>> 1/Inf = 0

>> 0/0 = NaN

These are NOT round off errors

Example of round off error :

```
>> format long
>> x = 9.4
>> y = 9.4 - 9
>> z = y - 0.4
```

non-zero !!

Application example: (Solve quadratic equation)

$$ax^2 + bx + c = 0$$

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$ formula for the roots of a quadratic equation

Suppose : $\boxed{a = 1,\ b = 9^{12},\ c = -3}$

$$x_\pm = \frac{-9^{12} \pm \sqrt{9^{24} + (4 \times 3)}}{2}$$

Minus sign root : $x_- = -2.824 \times 10^{11}$

Plus sign root : $x_+ = \dfrac{-9^{12} + \sqrt{9^{24} + 12}}{2} > 0$

However, MATLAB returns $\underbrace{x_+ = 0}_{\text{absurd because zero is NOT a root } (\because -3 \neq 0)}$

absurd because zero is NOT a root ($\because -3 \neq 0$)

How to fix the precision issue for x_+ :

$$x_+ = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$$

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

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

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

$$= \underbrace{\frac{-2c}{b + \sqrt{b^2 - 4ac}}}_{\text{typing in MATLAB gives}}$$

$$x_+ = 1.062 \times 10^{-11}$$