L01: FP Problem

Goal: To see that computation on a computer can be inaccurate, even if the math is correct.

Floating-Point Blues

Suppose we need to compute the integral

$$I_n = \int_0^1 \frac{x^n}{x+\alpha}\,dx$$

For a given real number α and integer n, $n \geq 0$.

This is tough to do, except for this trick...

$$I_n = \int_0^1 \frac{x^n}{x+\alpha}\,dx$$

$$= \int_0^1 \frac{x^n + x^{n-1}\alpha - x^{n-1}\alpha}{x+\alpha}\,dx$$

$$= \int_0^1 x^{n-1}\frac{x+\alpha}{x+\alpha} - \alpha\frac{x^{n-1}}{x+\alpha}\,dx$$

$$= \int_0^1 x^{n-1}\,dx - \alpha\int_0^1 \frac{x^{n-1}}{x+\alpha}\,dx$$

$$= \frac{1}{n} - \alpha I_{n-1} \quad \text{WOW!}$$

Thus, $I_n = \frac{1}{n} - \alpha I_{n-1}$ (recurrence relation)

Notice that I_0 is easy

$$I_0 = \int_0^1 \frac{1}{x+\alpha}\,dx = \ln(x+\alpha)\Big|_0^1 = \ln(1+\alpha) - \ln\alpha = \ln\frac{1+\alpha}{\alpha}$$

Cool! Let's try it out.

Create a Matlab script (text file with extension .m).

Comments → { % Try alpha values of 0.5 and 2.

Initialize params → { alpha = 0.5; N = 100;

I_0 → I = log((1+alpha) / alpha);

$I_n = \frac{1}{n} - \alpha I_{n-1}$ {

```
% Try alpha values of 0.5 and 2.
alpha = 0.5;
N = 100;

I = log((1+alpha) / alpha);

for n = 1:N
   I = 1/n - alpha * I;
end

disp(['Answer: ' num2str(I)]);
```

Print result →

For $\alpha = 0.5 \Rightarrow$ answer $= 0.0066444$

For $\alpha = 2 \Rightarrow$ answer $= 6.058 \times 10^{12}$

Hmmm... seems strange.

Observation: If $0 \le x \le 1$ and $\alpha > 1$, then $\frac{x^n}{x+\alpha} \le x^n$

Hence, $I_n = \int_0^1 \frac{x^n}{x+\alpha}\,dx \le \int_0^1 x^n = \frac{1}{n+1}$

So, for $\alpha=2$, we should get $I_{100} \le \frac{1}{101}$.

> Note: Aritmetic on a computer uses truncated numbers.
> Thus, we can have a small error in every number.

Thus, $I_0^{(comp)} = I_0^{(exact)} + e_0$

tiny error

and $I_n^{(comp)} = I_n^{(exact)} + e_n$

error at step n

Using our recurrence relation,

Using our recurrence relation,

$$I_n^{(exact)} = \frac{1}{n} - \alpha I_{n-1}^{(exact)} \quad \text{(mathematical)}$$

$$I_n^{(comp)} = \frac{1}{n} - \alpha I_{n-1}^{(comp)} \quad \text{(computational)}$$

Then, $e_n = I_n^{(comp)} - I_n^{(exact)}$

$$= \left(\frac{1}{n} - \alpha I_{n-1}^{(comp)}\right) - \left(\frac{1}{n} - \alpha I_{n-1}^{(exact)}\right)$$

$$= -\alpha \left(I_{n-1}^{(comp)} - I_{n-1}^{(exact)}\right)$$

$$e_n = -\alpha e_{n-1}$$

That is, $e_n = \alpha^2 e_{n-2}$

$$= \alpha^3 e_{n-3}$$

$$= \vdots$$

$$= \alpha^n e_0$$

$$\Rightarrow |e_n| = |\alpha|^n |e_0|$$

If $|\alpha| < 1 \Rightarrow |e_n| \to 0$ as $n \to \infty$ (Good)

If $|\alpha| > 1 \Rightarrow |e_n| \to \infty$ as $n \to \infty$ (Bad)

So there seems to be a build-up of round-off errors, but only when $|\alpha| > 1$.

Another example:

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \cdots$$

Suppose we use only 5 digits of accuracy.

$$e^{-5.5} = 1 - 5.5 + 15.125 - 27.729 + \cdots \quad (25 \text{ terms})$$

$$= 0.0026363$$

Mathematically, it's equivalent to

$$\frac{1}{e^{5.5}} = \frac{1}{1 + 5.5 + 15.125 + 27.729 + \cdots}$$

$$= \boxed{0.0040865}$$

It's not just what you compute, but how you compute it.
Consider adding up these 4 binary numbers, but keeping only 4 significant digits.

Method 1

0.1111
0.0111 → ⊕ 1.0110 = 0.1011 × 10 → ⊕ 0.1100 × 10 → ⊕ 0.1100 × 10
0.0011 → = 0.0001 × 10 (!)
0.0001 → 0.0000 × 10

Answer = 1.1

Method 2

0.0001
0.0011 → ⊕ 0.0100 → ⊕ 0.1011 (!) → ⊕ 1.101
0.0111
0.1111

Answer = 1.101

Take-Home Message

We follow some basic rules when doing arithemetic and mathematics. For example:

1) $(a+b)+c = a+(b+c)$
2) $a+e = a \Rightarrow e=0$
3) $\frac{a+b}{c} = \frac{a}{c} + \frac{b}{c}$
4) Correct mathematical algorithms produce correct answers.

Once you do arithmetic using floating-point numbers, none of the above are true.