Lecture: Numerical Analysis

- We are always dealing with finite memory
- How do we store numbers and compute on them.

Integers:

4 bit:

1	0	1	0

$= 1 \cdot 2^3 + 0 \, 2^2 + 1 \cdot 2^1 + 0 \cdot 2^1 = 10$

So we can represent #s from $0 - 2^k - 1$ with k bits

For signed integers we use a "sign" bit

1	1	0	1

$= 1(-1) + 1(2)^2 + 0(2)^1$

$+ 1(2)^{\sim} = -5$

In practice this would have two zeros. So instead we use 2s-complement.

$\boxed{\cancel{1} \mid 0 \mid 1 \mid 1 \mid 1}$

We can represent -2^{k-1} to $2^{k-1} - 1$

In modern architectures one integer takes 4 bytes = 32 bits

<u>Reals</u>: Still have limited memory and want to optimize <u>range</u> & <u>precision</u>.

We represent a real number by using a floating point representation

$$\pm \, d_0 . d_1 d_2 \cdots d_p \times b^{e} \leftarrow \text{exponent.}$$

sign

Significance, mantista

base (usually $b=2$)

We have to specify the range of $e \in (e_{min}, e_{max})$

±	$d_0 \; d_1 \; \cdots \; d_p$	e

sign

didgets in the significance

exponent

According to IEEE.

Precesion	Size	p	Bits in e
Single	32	24	8
Double	64	53	1

E[illegible] $1.0[illegible] \times 2^3 = 10$

Ex: $1.010 \times 2 = 10$

$$1.001 \times 2^2 = 1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 + 1 \cdot 2^{-1}$$

$$= 4.5$$

Ex: $p = 4, \; b = 2, \; 8 + 7.5 = \sigma$

$$8 = 1.000 \times 2^3 = 1.000 \times 2^3$$

$$7.5 = 1.111 \times 2^2 = 0.111$$

drop last bit b/c $p=4$

$$8 + 7.5 = 1.111 \times 2^3$$

$$= 8 + 4 + 2 + 1 = 15$$

So absolute error is

$$|\sigma - \tilde{\sigma}| = |15 - 15.5| = 0.5$$

and relative error is

$$\frac{|\sigma - \tilde{\sigma}|}{|\tilde{\sigma}|} = \frac{0.5}{15.5} \approx 0.03$$

$\boxed{\text{Take away:}}$ Adding numbers of sig. different magnitude leads to precesion error because of dropped bits.

Ex: $8 - 7.5 =$

$$\begin{array}{r} 1.000 \times 2^3 \\ -\,0.111 \times 2^3 \\ \hline 0.001 \times 2^3 \end{array} = 1$$

Abs error: $|1 - 0.5| = 0.5$

Rel error: $\frac{0.5}{0.5} = 1$ "catastrophic cancellation"

$\boxed{\text{Takeaway:}}$

(i) Subtracting things of similar magnitudes is bad.

(ii) Be careful with overflows & underflows.

(iii) Be careful when comparing floating point numbers

i.e. never do $x == 0$

instead do $|x-0| < \epsilon$

There are "special" floating point #s

- Zero: $e = e_{min} - 1$
- infinity: $e = e_{max} + 1$ and Significand $= 0$
- NaN = "not a number"

$e = e_{max} + 1$ Significand $\neq 0$

e.g. $\frac{0}{0}$, $\log(-1)$