

* Idag: Flyttal,
avrundningsfel,
(feluppskattningar)
(orientering)

713

"Anteckningar, avsnitt 1.6"

* Datorrepresentation av reella tal:
"Flyttal"

- Reella tal: oändligt många decimaler
- Flyttal: ändligt många decimaler

$\Rightarrow$ Flyttal är alltid rationella tal!

$\Rightarrow$ Avrundningsfel

Viktigt att känna till vid numeriska beräkningar!

* Representation av flyttal

Skrivs på exponentform:

$$x = m \cdot b^e$$

$\uparrow$ mantissa $\uparrow$ bas $\leftarrow$ exponent

Normaliserad mantissa:

$$1 \leq |m| < b$$

$\cap \mathbb{Q}$

Bas:

$$b = 10 \quad (\text{decimalt talssystem})$$

$$b = 2 \quad (\text{binärt talssystem})$$

$$b = 16 \quad (\text{hexadecimalt talssystem})$$

Exponent:

$$e_{\min} \leq e \leq e_{\max}$$

$\cap \mathbb{Z}$

Exempel: ($b=10$)

$$\begin{aligned}
 X &= 3.1415 \\
 &= \underbrace{3.1415}_{1 \leq |m| < 10} \cdot \underbrace{10^0}_{b \leftarrow e}
 \end{aligned}$$

$$\begin{aligned}
 X &= 314159265 \\
 &= \underbrace{3.14159265}_{1 \leq |m| < 10} \cdot \underbrace{10^4}_{b \leftarrow e}
 \end{aligned}$$

Exempel: ($b=2$)

$$\begin{aligned}
 X &= 1.011001 \\
 &= \underbrace{1.011001}_{1 \leq |m| < 2} \cdot \underbrace{2^0}_{b \leftarrow e}
 \end{aligned}$$

$$\begin{aligned}
 X &= 1011001.11011 \\
 &= \underbrace{1.01100111011}_{1 \leq |m| < 2} \cdot \underbrace{2^6}_{b \leftarrow e}
 \end{aligned}$$

Notera: De enda siffror som finns
är $0, \dots, b-1$, dvs
 0 och 1

* Något om talsystem

På samma sätt som

$$\begin{aligned}
 255_{10} &= 2 \cdot 100 + 5 \cdot 10 + 5 \cdot 1 \\
 &= 2 \cdot 10^2 + 5 \cdot 10^1 + 5 \cdot 10^0
 \end{aligned}$$

har vi

$$\begin{aligned}
 11111111_2 &= 1 \cdot 2^7 + 1 \cdot 2^6 + 1 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 \\
 &= 128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 \\
 &= 255
 \end{aligned}$$

$$\therefore 255_{10} = 11111111_2$$

Allmänt:

$$(a_n a_{n-1} a_{n-2} \dots a_1 a_0)_b = \sum_{k=0}^n a_k \cdot b^k$$

Notera att speciellt gäller

$$\underbrace{(b-1)(b-1)\dots(b-1)}_{n \text{ siffror}} = b^n - 1 \quad (k=0,1,2,\dots,n-1)$$

Övning: Visa detta!

Ledning: Använd välbekant lemma från F14...

Exempel:

$$11111111_2 = 2^8 - 1 = 256 - 1 = 255$$

$$999_{10} = 10^3 - 1 = 1000 - 1 = 999$$

* Andra talsystem:

triära ($b=3$)

kvartenära ($b=4$)

:

hexadecimala ($b=16$)

↑ 0, 1, 2, 3, ..., 9, A, B, C, D, E, F

* Exempel: RGB (röd-grön-blå)
(Färgpalett för HTML och grafikprogram)

$$\underbrace{FF}_{R} \underbrace{FF}_{G} \underbrace{FF}_{B} = \begin{cases} \text{Röd} & FF = 15 \cdot 16 + 15 \cdot 1 = 255 \\ \text{Grön} & FF_{16} = 255 \\ \text{Blå} & FF_{16} = 255 \end{cases}$$

= "vit"

Varje färg får variera mellan

0 och $FF_{16} = 255$. Ger

$$256 \cdot 256 \cdot 256 = (2^8)^3 = 2^{24}$$

$$= 2^{10+10+4} = 2^{10} \cdot 2^{10} \cdot 2^4$$

$$= 1024$$

$$\approx 1000 \cdot 1000 \cdot 16 = 16 \text{ miljoner}$$

olika färger.

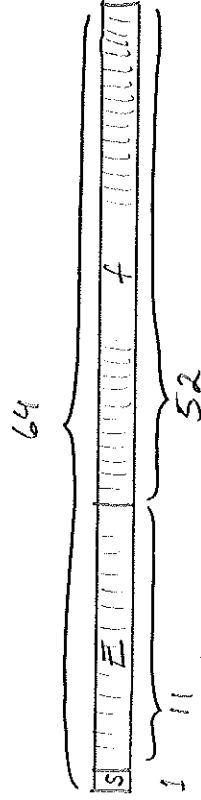
* Flyktstandarden IEEE 754

Två versioner:

Binary32 ← "float" i C/C++

Binary64 ← "double" i C/C++
MATLAB

- Bas 2
- 64 bitar / tal (Binary64), dvs 8 bytes
(1 byte = 8 bitar)
- $m = 1.f$, 1 lagras ej
- 1 stället för e lagras $E = e + 1027$
- En bit används för att lagra tecknet
- 11 bitar används för att lagra E
- 52 bitar används för att lagra f



Största tal: $\approx \pm 2 \cdot 10^{308}$

Minsta tal: $\approx \pm 2 \cdot 10^{-308}$

$\epsilon_{mach} \approx 2 \cdot 10^{-16}$

* "machine precision"

* Machine precision, avrundningsfel

- Alla tal representeras med ett relativt fel som är som störst ϵ_{mach} :

$$fl(x) = x \cdot (1 + \Delta)$$

$$|\Delta| \leq \epsilon_{mach}$$

- Alla operationer har ett relativt fel som är som störst ϵ_{mach} :

$$fl(fl(x) \cdot fl(y)) = fl(x) \cdot fl(y) \cdot (1 + \Delta)$$

$$|\Delta| \leq \epsilon_{mach}$$

* Exempel:

$$\text{Beräkna } f(x) = \frac{4x}{(1+x)^2 - (1-x)^2}$$

Notera först att

$$f(x) = \frac{4x}{1+x^2+2x - (1+x^2-2x)} = \frac{4x}{4x} = 1$$

MATLAB ger

$$f(1) = 1$$

$$f(0.001) = 1.00 \dots 08 \frac{7}{7}$$

$$f(10^{-10}) = 0.99 \dots 172$$

$$f(10^{-16}) = 1.8014 \dots$$

$$f(10^{-17}) = 1.4f$$

Om $|x|$ är litet gäller att

$$\frac{4x}{(1+x)^2 - (1-x)^2} \approx \frac{4x}{1+2x - (1-2x)}$$

MATLAB (IEEE 754) räknar

$$\frac{4x}{(1+2x) \cdot (1+\Delta_1) - (1-2x) \cdot (1+\Delta_2)}$$

$$= \frac{4x}{1 + \Delta_1 + 2x + \underbrace{2x\Delta_1}_{\approx 0} - (1 + \Delta_2 - 2x - \underbrace{2x\Delta_2}_{\approx 0})}$$

$$\approx \frac{4x}{4x + \Delta_1 + \Delta_2}$$

Om $x \sim \Delta_1 \sim \Delta_2$ så kan resultatet bli nästan vad som helst...

Demo: (MATLAB)

- Beräkning av $f(x) = \frac{4x}{(1+x)^2 - (1-x)^2}$

- Beräkning av största/minsta tal
(notera "subnormala" tal)

- Beräkning av ϵ_{mach}

- Mandelbrot / fixpunktiteration