

Část I

Strojová čísla a numerická matematika

10 Numerická matematika

10.1 Co to je?

Numerická matematika

Numerická matematika se věnuje matematickým metodám hledajícím přibližná řešení matematických úloh a jejich spolehlivosti.

Zahrnuje například metody pro...

1. řešení soustav lineárních rovnic,
2. řešení (obyčejných i parciálních) diferenciálních rovnic,
3. výpočet integrálů,
4. vyhodnocování funkčních hodnot,
5. odhadování chyb při výpočtech,
6. hledání lokálních a globálních extrémů (optimalizační úlohy),
7. výpočet vlastních čísel a vlastních vektorů,
8. faktorizace matic,
9. ...

Typicky k řešení úloh využívá počítačů.

Z dějin neúspěchu...

- Chyba v raketě [Patriot](#) (28 mrtvých)

$$(0.1)_{10} = (0.00011001100110011001100110011 \dots)_2$$

- Exploze rakety [Ariane 5](#) konverze z 64-bitového čísla s plovoucí čárkou na 16-bitové celé se znaménkem. (\$7 miliard na vývoj, raketa a náklad za půl miliardy.)



přesnost	délka m	$d =$ délka e	parametr b
poloviční (binary16, <i>half precision</i>)	10	5	15
jednoduchá (binary32, <i>single precision</i>)	23	8	127
dvojitá (binary64, <i>double precision</i>)	52	11	1023
čtyřnásobná (binary128, <i>quadruple prec.</i>)	112	15	16383

Reprezentovaná hodnota x se pak určí následujícím způsobem:

- pokud $e = 2^d - 1$ a $m \neq 0$, tak $x = \text{NaN}$ (*Not a Number*)
- pokud $e = 2^d - 1$ a $m = 0$, tak $x = (-1)^s \cdot \text{Inf}$
- pokud $0 < e < 2^d - 1$, pak $x = (-1)^s \cdot (1.m_2)_2 \cdot 2^{e-b}$ (tzv. **normalizovaná čísla**)
- pokud $e = 0$ a $m \neq 0$, pak $x = (-1)^s \cdot (0.m_2)_2 \cdot 2^{1-b}$ (tzv. **subnormální čísla**)
- pokud $e = 0$ a $m = 0$, pak $x = (-1)^s \cdot 0$

Skrytá jednička

Všimněme si, že pro normalizované číslo $x = (-1)^s \cdot (1.m_2)_2 \cdot 2^{e-b}$ jsme vlastně uložili o 1 platnou cifru více, než kolik je délka m . (Tedy např. v jednoduché přesnosti uložíme 24 platných cifer.)

Této konvenci se říká různě: skrytá jednička, skrytý bit, vedoucí bit, implicitní bit.

Strojová čísla (1/3)

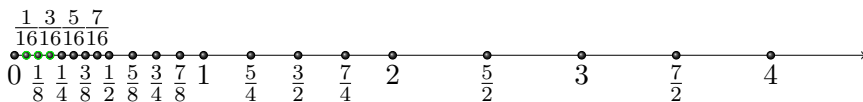
Reálná čísla, která lze reprezentovat popsáním způsobem, se nazývají **strojová čísla**.

Příklad: Vezměme dvoubitové m , exponent e s třemi bity (tj. $d = 3$) a $b = 3$.

Dostaneme následující množinu strojových čísel (vypisujeme jen nezáporná):

$$\left\{ 0, \frac{1}{16}, \frac{1}{8}, \frac{3}{16}, \frac{1}{4}, \frac{5}{16}, \frac{3}{8}, \frac{7}{16}, \frac{1}{2}, \frac{5}{8}, \frac{3}{4}, \frac{7}{8}, 1, \frac{5}{4}, \frac{3}{2}, \frac{7}{4}, 2, \frac{5}{2}, 3, \frac{7}{2}, 4, 5, 6, 7, 8, 10, 12, 14 \right\}$$

Subnormální strojová čísla jsou zvýrazněna **zeleně**.



Množina všech strojových čísel s danou přesností (tj. specifikací m , e a b) nemá nic moc společného s reálnými čísly. Jde o konečnou podmnožinu racionálních čísel.

Strojová čísla (2/3)

Označme množinu strojových čísel symbolem $F \equiv F(|m|, |e|, b)$. Tímto zápisem zdůrazňujeme, že F závisí na zvoleném počtu bitů pro m , e a parametru b .

Množina F , jakožto konečná podmnožina \mathbb{R} má samozřejmě největší a nejmenší prvek ($\min F$ a $\max F$). Dále jsou zajímavé následující hodnoty:

přesnost	max. č.	min. kladné norm. č.	min. kladné subnorm. č.
single	$(2 - 2^{-23})2^{127}$ $\approx 3.4 \cdot 10^{38}$	2^{-126} $\approx 1.2 \cdot 10^{-38}$	$2^{-126-23} = 2^{-149}$ $\approx 1.4 \cdot 10^{-45}$
double	$(2 - 2^{-52})2^{1023}$ $\approx 1.8 \cdot 10^{308}$	2^{-1022} $\approx 2.2 \cdot 10^{-308}$	$2^{-1022-52} = 2^{-1074}$ $\approx 4.9 \cdot 10^{-324}$

Skutečně, pro **jednoduchou** přesnost máme pro m 23 bitů, e 8 bitů a $b = 127$. Proto

$$\max F = (1.1 \dots 1)_2 \cdot 2^{254-127} = \frac{1 - (1/2)^{24}}{1 - 1/2} \cdot 2^{127} = (2 - 2^{-23}) \cdot 2^{127}$$

$$\text{minimální kladné normalizované} = (1.0 \dots 0)_2 \cdot 2^{1-127} = 2^{-126},$$

$$\text{minimální kladné subnormální} = (0.0 \dots 1)_2 \cdot 2^{1-127} = 2^{-23-126} = 2^{-149}.$$

Strojová čísla (3/3)

F je charakterizováno pomocí **strojové přesnosti** ε_F (*machine epsilon*), což je vzdálenost čísla $1 = +1 \cdot 2^0$ od nejbližšího většího čísla v F , tj.

$$\varepsilon_F = (1.0 \dots 01)_2 \cdot 2^0 - (1.0 \dots 00)_2 \cdot 2^0.$$

Pro jednoduchou přesnost proto platí $\varepsilon_F = 2^{-23}$ a pro dvojitou $\varepsilon_F = 2^{-52}$.

Tvrzení 11.1. Vzdálenost libovolného normalizovaného čísla $x \in F$ od jeho nejbližších sousedů z F je nejméně $\varepsilon_F \frac{|x|}{2}$ a nejvíce $\varepsilon_F |x|$.

Reprezentace reálných čísel (1/3)

Nechť $fl : \mathbb{R} \rightarrow F$ je zobrazení, které přiřadí každému $x \in \mathbb{R}$ „nejbližší“ strojové číslo. (Zkratka fl je od slova „float“.)

„Nejbližší“ je určeno podle vybrané strategie zaokrouhlování (k nejbližšímu, k \pm nekonečnu, náhodně)² či usekávání (zaokrouhlování směrem k nule)³.

Při pokusu o reprezentaci čísel mimo rozsah dochází k **přetečení** (*overflow*) respektive **podtečení** (*underflow*).

Definice 11.2 Nechť číslo $\alpha \in F$ je přibližnou hodnotou čísla $a \in \mathbb{R}$.

- **Absolutní chybou** reprezentace a pomocí α rozumíme hodnotu $|\alpha - a|$.
- Pro $a \neq 0$ je **relativní chyba** reprezentace a pomocí α rovna

$$\frac{|\alpha - a|}{|a|}.$$

²Round to nearest, ties to even; Round to nearest, ties away from zero; Round towards +infinity; Round towards -infinity; Stochastic rounding

³Round towards zero

Reprezentace reálných čísel (2/3)

Uvažujme reálné číslo x takové, že lze psát

$$x = q \cdot 2^\ell, \quad \text{kde } 1 \leq q < 2 \text{ a } -126 \leq \ell \leq 126.$$

Hrubě řečeno, x je v rozsahu normalizovaných čísel v jednoduché přesnosti.

Jaká je **chyba** vzniknuvší při zaokrouhlení na nejbližší strojové číslo?

Pro jednoduchost budeme *zaokrouhlovat směrem k nule*, tedy usekneme bity přesahující délku signifikandy (x je kladné). Necht

$$x = (1.m_1m_2m_3m_4\dots)_2 \cdot 2^\ell,$$

pak

$$\text{fl}(x) = (1.m_1m_2\dots m_{23})_2 \cdot 2^\ell,$$

Pro absolutní chybu platí $|x - \text{fl}(x)| \leq 2^{-23+\ell}$ a pro relativní chybu

$$\frac{|x - \text{fl}(x)|}{|x|} \leq \frac{2^{-23+\ell}}{q \cdot 2^\ell} \leq 2^{-23}.$$

Reprezentace reálných čísel (3/3)

Této mezi pro relativní chybu se říká **zaokrouhlovací jednotka** (*unit roundoff error*) a značí se $\mathbf{u} = 2^{-23}$.

Pozor, tato definice není ustálená a je zaměňována s výše uvedenou strojovou přesností (s různými variantami detailů).

Pokud bychom použili *zaokrouhlování směrem k nejbližšímu*⁴, dostaneme $\mathbf{u} = 2^{-24}$.

Tvrzení 11.3. *Necht $x \in \mathbb{R}$ leží mezi největším a nejmenším normalizovaným kladným číslem množiny F . Pak platí*

$$\text{fl}(x) = x(1 + \delta), \quad \text{kde } |\delta| \leq \mathbf{u}.$$

Poznámka: δ zde závisí na x , tj. $\delta = \delta(x)$. Například pro strojové číslo x je $\delta = 0$.

11.2 Aritmetické operace

Aritmetické operace a chyba při jejich provádění

Se strojovými čísly můžeme provádět obvyklé základní číselné operace, např.: $+$: $(x, y) \mapsto \text{fl}(x + y)$.

Z předchozího tvrzení o zaokrouhlování ihned plyne:

Tvrzení 11.4. *Necht $x, y \in F$ a \odot značí operaci sčítání, odečítání, násobení nebo dělení. Pokud nedojde k přetečení nebo podtečení (zůstali jsme v intervalu normalizovaných čísel), tak platí*

$$\text{fl}(x \odot y) = (x \odot y)(1 + \delta), \quad \text{kde } |\delta| \leq \mathbf{u}.$$

⁴nezáleží, jak si poradíme s nerozhodnutelnými právě mezi 2 nejbližšími

- Uvědomme si, že sčítáním/odečítáním/násobením/dělením dvou strojových čísel nemusíme nutně dostat opět strojové číslo! Obecně jde o reálné číslo, které je potřeba zaokrouhlit. (Tedy nemusí platit $\delta = 0$.)
- Model je příliš jednoduchý pro dnešní procesory, některé např. umí FMA: Fused Multiply Add, které počítá $(x, y, z) \mapsto x \pm yz$ s jedním zaokrouhlením.

Aritmetické operace – katastrofická ukázka 1/2

Mějme funkci $f : \mathbb{R}^2 \rightarrow \mathbb{R}$ definovanou takto:

$$f(x, y) = 333.75y^6 + x^2 \left(11x^2y^2 - y^6 - 121y^4 - 2 \right) + 5.5y^8 + \frac{x}{2y}.$$

Vyhodnotíme $f(77617, 33096)$ v různých přesnostech

Aritmetické operace – katastrofická ukázka 2/2

přesnost	$f(77617, 33096)$
SageMath (přesnost 23 bitů)	1.17260
SageMath (přesnost 24 bitů)	$-6.33825 \cdot 10^{-29}$
SageMath (přesnost 53 bitů)	$-1.18059162071741 \cdot 10^{21}$
SageMath (přesnost 54 bitů)	$1.18059162071741 \cdot 10^{21}$
SageMath (přesnost 100 bitů)	1.1726039400531786318588349045
SageMath (přesnost 121 bitů)	1.17260394005317863185883490452018371
SageMath (přesnost 122 bitů)	-0.827396059946821368141165095479816292

Přesný výsledek je $f(77617, 33096) = -\frac{54767}{66192} \approx -0.827396$.

[S. M. Rump: *Algorithms for verified inclusions – theory and practice*, 1988]

Ztráta platných cifer (1/3)

Došlo ke kumulaci chyb při provádění aritmetických operací.

Velké problémy může způsobit tzv. **krácení** (*cancellation*), které ovšem na první pohled nemusí být patrné.

Předvedeme si jej na ilustrativním příkladě. Představme si, že počítač počítá v desítkové soustavě a zaokrouhluje na 10 platných cifer.

Chceme vyhodnotit výraz $x - \sin(x)$ pro $x = \frac{1}{15}$.

$$\begin{aligned} x &\leftarrow 6.6666\ 66667 \cdot 10^{-2} \\ \sin(x) &\leftarrow 6.6617\ 29492 \cdot 10^{-2} \\ x - \sin(x) &\leftarrow 0.0049\ 37175 \cdot 10^{-2} \\ x - \sin(x) &\leftarrow 4.9371\ 75000 \cdot 10^{-5} \end{aligned}$$

Poslední 3 *nuly* nejsou *správné* platné cifry. Během výpočtu jsme o ně přišli.

Spočítejme relativní chybu našeho výpočtu...

Ztráta platných cifer (2/3)

Relativní chyba výpočtu je

$$\frac{\left| \frac{1}{15} - \sin\left(\frac{1}{15}\right) - \text{fl}\left(\text{fl}\left(\frac{1}{15}\right) - \sin\left(\text{fl}\left(\frac{1}{15}\right)\right)\right)\right|}{\left| \frac{1}{15} - \sin\left(\frac{1}{15}\right) \right|} \approx 1.4 \cdot 10^{-7}.$$

To je hodně v porovnání se zaokrouhlovací jednotkou v této aritmetice

$$\frac{|x - \text{fl}(x)|}{|x|} \leq 5 \cdot 10^{-10} = \mathbf{u},$$

kde x je v rozsahu normalizovaných čísel.

Tvrzení 11.5. *Nechť x a y jsou normalizovaná strojová čísla a platí $x > y > 0$. Pokud $2^{-p} \leq 1 - \frac{y}{x} \leq 2^{-q}$ pro nějaká kladná celá p a q , tak platí, že nejvíce p a nejméně q platných binárních bitů je ztraceno při provedení odečítání $x - y$.*

Ztráta platných cifer (3/3)

Krácení se lze vyhnout několika technikami:

- přeformulováním problému tak, aby nedocházelo k odečítání,
- použitím rozvoje funkcí do řad (např. do Taylorovy řady),
- použitím jiných rovností ...
- (použitím přesné aritmetiky)

Další čtení

[Základní potíže při práci s čísly s plovoucí čárkou I](#)

[Základní potíže při práci s čísly s plovoucí čárkou II](#)

[Implementace funkce sinus v libm](#)

11.3 Závěr

Zaokrouhlovací chyby – shrnutí

Původ zaokrouhlovacích chyb:

- zaokrouhlovací chyby jednotlivých operací a jejich kumulace,
- krácení.

Několik poznámek k zaokrouhlovacím chybám:

- zvýšení přesnosti nemusí dát přesnější výsledek,
- krácení může být někdy výhodné – lze tak vyrušit zaokrouhlovací či jiné chyby,
- málo operací s malými čísly neznamená, že chyba bude malá.

Nebereme v úvahu hardware (např x87 vs SSE2, FMA...).

Zaokrouhlovací chyby – alternativní přístupy

Jeden z **problémů** strojových čísel (IEEE-754 apod.) spočívá v ignoraci vznikuvší chyby při výpočtu.

Možné alternativy (stručně):

- Použít exaktní aritmetiku \mathbb{Z} , \mathbb{Q} či $GF(p)$ (není vždy možné a vhodné).
- **Intervalová aritmetika** (místo jednoho strojového čísla pracujeme s dvěma reprezentujícími krajní body intervalu, jehož délka představuje neurčitost ve znalosti příslušného „čísla“). (IEEE 1788–2015)
- **Unum**.
- ...

ChangeLog

Verze	Datum	Autor	Log
1.41	15.11.2023	ŠS	Slajd o skryté jedničce.
1.4	23.11.2020	ŠS	Ladění: zdroje chyb obecně, binary16.
1.3	3.12.2019	ŠS	Odstranění nejasného pojmu matematické zaokrouhlování.
1.2	11.12.2019	ŠS	Upřesnění standardního modelu odhadu chyb.
1.1	3.12.2019	ŠS	Upřesnění strategií zaokrouhlování.
1.0	25.11.2019	ŠS	Výchozí verze.