Numerické metody

Doc. RNDr. Libor Čermák, CSc. RNDr. Rudolf Hlavička, CSc.

Ústav matematiky
Fakulta strojního inženýrství
Vysoké učení technické v Brně

6. února 2006

Obsah

Při řešení problémů reálného světa se stále častěji setkáváme s potřebou popsat zkoumanou skutečnost pomocí věrohodného matematického modelu a ten pak uspokojivě vyřešit. Žijeme v době počítačů a tak je přirozené, že k realizaci matematického modelu počítač využijeme. Počítače umí pracovat velmi rychle s informacemi kódovanými pomocí čísel.

A právě zde je místo pro numerickou matematiku (v angličtině numerical analysis) jakožto vědní disciplínu, která vyvíjí a analyzuje metody, jejichž „technologickým jádrem" jsou manipulace s čísly. V posledních letech se v anglicky psané literatuře místo termínu „numerical analysis" stále častěji používá termín scientific computing (odpovídající český termín nám bohužel není znám).

Když chceme metodami numerické matematiky vyřešit daný problém popsaný obecným matematickým modelem, musíme takový model nejdříve „digitalizovat", to jest formulovat ho ve tvaru numerické úlohy, jejíž vstupní i výstupní data jsou čísla. Numerická metoda je postup řešení numerické úlohy. Přesný popis kroků realizujících numerickou metodu označujeme jako algoritmus numerické metody. Lze ho vyjádřit jako posloupnost akcí (proveditelných na počítači), které k danému (přesně specifikovanému konečnému) souboru vstupních čísel jednoznačně přiřadí odpovídající (přesně specifikovaný konečný) soubor výstupních čísel.

Příprava rozsáhlých souborů vstupních dat bývá označována jako preprocessing. Rozsah souboru výsledných údajů je často ohromný, pro člověka „nestravitelný", a proto je třeba výsledky vhodně zpřístupnit tak, aby je zadavatel výpočtu byl vůbec schopen vyhodnotit. Metodám, které to provádějí, se říká postprocessing. Jednou z forem postprocessingu je vizualizace výsledků.

Jako příklad „problému ze života" uvažujme předpověď počasí. Pohyb vzduchu v atmosféře dovedeme alespoň přibližně popsat pomocí soustav parciálních diferenciálních rovnic a vhodných doplňujících podmínek. Metodami numerické matematiky dokážeme tyto rovnice přibližně řešit. Potřebná vstupní data se získávají pomocí družic a pozemních meteorologických stanovišť. Výsledky numerických výpočtů zpracované do animovaných meteorologických map pak sledujeme v televizní předpovědi počasí.

Při řešení reálných problémů téměř nikdy nezískáme přesné řešení, musíme se spokojit jen s řešením přibližným, které je zatíženo chybami. Naším cílem je organizovat výpočet tak, aby celková chyba byla co nejmenší.

Především se musíme vyvarovat hrubých lidských chyb, které vyplývají z nepochopení problému a z nepozornosti nebo nedbalosti člověka při jeho řešení.

Chyba matematického modelu. Při vytváření matematického modelu reálného problému provádíme vždy jisté idealizace. Rozdíl mezi řešením idealizovaného problému a řešením problému reálného nazýváme chybou matematického modelu. Do této kategorie chyb zahrnujeme také chyby ve vstupních údajích.

Příklad. Máme určit povrch zemského pláště. K výpočtu použijeme vzorec $S = 4\pi r^2$ pro povrch koule o poloměru r. Chyba modelu spočívá v předpokladu, že Země je koule.

Chyba numerické metody. Jestliže k řešení (numerické) úlohy použijeme numerickou metodu, která nám neposkytne přesné (teoretické) řešení dané úlohy, pak chybu, které se dopustíme, nazýváme chybou numerické metody. Důležitou součástí návrhu numerické metody je odhad chyby numerické metody.

Příklad. Máme spočítat hodnotu funkce $\sin 1$ sečtením konečného počtu členů Taylorovy řady

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} - \cdots + (-1)^n \frac{x^{2n+1}}{(2n+1)!} + \ \cdots$$

pro $x = 1$. Je známo, že sečtením prvních tří členů řady se dopustíme chyby velikosti nejvýše $1/7!$, obecně sečtením prvních n členů se dopustíme chyby nejvýše $1/(2n+1)!$.

Zaokrouhlovací chyby. Při práci na počítači můžeme k reprezentaci čísel použít jen konečný počet cifer. Pracujeme proto s přibližnými hodnotami čísel, které dostaneme zaokrouhlením přesných hodnot. Zaokrouhlovací chyby vznikají už při vkládání dat do počítače, další pak vznikají při číselných výpočtech. Při špatně organizovaném výpočtu může dojít v důsledku nahromadění zaokrouhlovacích chyb k naprostému znehodnocení výsledku, viz příklad 1.7.

Příklad. Číslo π neumíme do počítače vložit přesně. Také výsledek operace, při níž číslo 2 dělíme číslem 3, nezobrazíme na standardním počítači pracujícím s binárními čísly přesně.

Je třeba mít na paměti, že při řešení reálného problému vystupují obvykle všechny chyby současně.

Obsah

Absolutní a relativní chyba. Ve výpočtech jsme často nuceni nahradit přesné číslo x přibližným číslem $\tilde{x}$. Číslo $\tilde{x}$ potom nazýváme aproximací čísla x. Rozdíl $\tilde{x} - x = \Delta x$ nazýváme absolutní chybou aproximace $\tilde{x}$ a číslo

$$\frac{\Delta x}{x} = \frac{\tilde{x} - x}{x}, \qquad x \neq 0,$$

nazýváme relativní chybou aproximace $\tilde{x}$. Pro $|\Delta x| \leq \varepsilon$ se používá také symbolický zápis $x = \tilde{x} \pm \varepsilon$ a míní se tím, že $\tilde{x} - \varepsilon \leq x \leq \tilde{x} + \varepsilon$. Podobně se pro $|\Delta x / x| \leq \delta$ používá zápis $\tilde{x} = x(1 \pm \delta)$. Absolutní hodnota relativní chyby se často uvádí v procentech.

Nyní posoudíme chybu, které se dopustíme při výpočtu hodnoty $f(x_1, x_2, \ldots, x_n)$ funkce f, když přesné hodnoty x_i nahradíme přibližnými hodnotami $\tilde{x}_i = x_i + \Delta x_i$.
Z Taylorova rozvoje $f(\tilde{\mathbf{x}})$ okolo bodu $\mathbf{x}$ dostaneme

$$f(\tilde{\mathbf{x}}) = f(\mathbf{x}) + \sum_{i=1}^{n} \Delta x_i \frac{\partial f(\mathbf{x})}{\partial x_i} + \frac{1}{2} \sum_{i,j=1}^{n} \Delta x_i \Delta x_j \frac{\partial^2 f(\mathbf{x})}{\partial x_i \, \partial x_j} + \cdots .$$

Považujeme-li součiny chyb $\Delta x_i \Delta x_j$ za malé, máme pro absolutní chybu

$$|\Delta f(\mathbf{x})| := |f(\tilde{\mathbf{x}}) - f(\mathbf{x})| \doteq \left| \sum_{i=1}^{n} \frac{\partial f(\mathbf{x})}{\partial x_i} \Delta x_i \right| \lesssim \sum_{i=1}^{n} \left| \frac{\partial f(\mathbf{x})}{\partial x_i} \right| \cdot |\Delta x_i| \,, \qquad (1.1)$$

Nyní posoudíme chybu, které se dopustíme při výpočtu hodnoty $f(x_1, x_2, \ldots, x_n)$ funkce f, když přesné hodnoty x_i nahradíme přibližnými hodnotami $\tilde{x}_i = x_i + \Delta x_i$. Z Taylorova rozvoje $f(\tilde{\mathbf{x}})$ okolo bodu $\mathbf{x}$ dostaneme

$$f(\tilde{\mathbf{x}}) = f(\mathbf{x}) + \sum_{i=1}^{n} \Delta x_i \frac{\partial f(\mathbf{x})}{\partial x_i} + \frac{1}{2} \sum_{i,j=1}^{n} \Delta x_i \Delta x_j \frac{\partial^2 f(\mathbf{x})}{\partial x_i \, \partial x_j} + \cdots .$$

Považujeme-li součiny chyb $\Delta x_i \Delta x_j$ za malé, máme pro absolutní chybu

$$|\Delta f(\mathbf{x})| := |f(\tilde{\mathbf{x}}) - f(\mathbf{x})| \doteq \left| \sum_{i=1}^{n} \frac{\partial f(\mathbf{x})}{\partial x_i} \Delta x_i \right| \lesssim \sum_{i=1}^{n} \left| \frac{\partial f(\mathbf{x})}{\partial x_i} \right| \cdot |\Delta x_i| \,, \qquad (1.1)$$

Nyní posoudíme chybu, které se dopustíme při výpočtu hodnoty $f(x_1, x_2, \ldots, x_n)$ funkce f, když přesné hodnoty x_i nahradíme přibližnými hodnotami $\tilde{x}_i = x_i + \Delta x_i$. Z Taylorova rozvoje $f(\tilde{\mathbf{x}})$ okolo bodu $\mathbf{x}$ dostaneme

$$f(\tilde{\mathbf{x}}) = f(\mathbf{x}) + \sum_{i=1}^{n} \Delta x_i \frac{\partial f(\mathbf{x})}{\partial x_i} + \frac{1}{2} \sum_{i,j=1}^{n} \Delta x_i \Delta x_j \frac{\partial^2 f(\mathbf{x})}{\partial x_i \, \partial x_j} + \cdots .$$

Považujeme-li součiny chyb $\Delta x_i \Delta x_j$ za malé, máme pro absolutní chybu

$$|\Delta f(\mathbf{x})| := |f(\tilde{\mathbf{x}}) - f(\mathbf{x})| \doteq \left| \sum_{i=1}^{n} \frac{\partial f(\mathbf{x})}{\partial x_i} \Delta x_i \right| \lesssim \sum_{i=1}^{n} \left| \frac{\partial f(\mathbf{x})}{\partial x_i} \right| \cdot |\Delta x_i| \,, \tag{1.1}$$

(symbol $\lesssim$ má význam „přibližně nerovno“) a pro chybu relativní

$$\left|\frac{\Delta f(\mathbf{x})}{f(\mathbf{x})}\right| \doteq \left|\sum_{i=1}^{n} \frac{x_i}{f(\mathbf{x})} \frac{\partial f(\mathbf{x})}{\partial x_i} \frac{\Delta x_i}{x_i}\right| \lesssim \sum_{i=1}^{n} \left|\frac{x_i}{f(\mathbf{x})} \frac{\partial f(\mathbf{x})}{\partial x_i}\right| \cdot \left|\frac{\Delta x_i}{x_i}\right| . \qquad (1.2)$$

Při praktických odhadech se hodnota funkce f a hodnoty jejích derivací $\partial f / \partial x_i$ na pravých stranách přibližných nerovností (1.2) a (1.1) počítají v bodě $\tilde{\mathbf{x}}$.

Chyby základních aritmetických operací. Zvolíme-li $f(x, y) = x \pm y$, dostaneme pomocí (1.1) a (1.2) pro absolutní a relativní chybu součtu a rozdílu

$$\Delta(x \pm y) \doteq \Delta x \pm \Delta y \,, \qquad \frac{\Delta(x \pm y)}{x \pm y} \doteq \frac{x}{x \pm y}\frac{\Delta x}{x} \pm \frac{y}{x \pm y}\frac{\Delta y}{y} \,. \tag{1.3}$$

Pro vyjádření chyby součinu volíme $f(x, y) = xy$ a obdržíme

$$\Delta(xy) \doteq y\Delta x + x\Delta y \,, \qquad \frac{\Delta(xy)}{xy} \doteq \frac{\Delta x}{x} + \frac{\Delta y}{y} \tag{1.4}$$

a pro chybu podílu dostaneme volbou $f(x, y) = x/y$

$$\Delta\left(\frac{x}{y}\right) \doteq \frac{1}{y}\Delta x - \frac{x}{y^2}\Delta y \,, \qquad \frac{\Delta(x/y)}{x/y} \doteq \frac{\Delta x}{x} - \frac{\Delta y}{y} \,. \tag{1.5}$$

Všimněte si, že relativní chyba součtu resp. rozdílu může být výrazně větší než relativní chyby operandů v případě, když $|x \pm y|$ je podstatně menší než $|x|$ nebo $|y|$. Při dělení malým číslem zase vzroste chyba absolutní.

Platné dekadické cifry. Nechť $\tilde{x}$ je aproximace čísla x, kterou zapišme v mocninném dekadickém rozvoji jako

$$\tilde{x} = \pm\,[\, d_1 \cdot 10^e + d_2 \cdot 10^{e-1} + \cdots + d_k \cdot 10^{e+1-k} + d_{k+1} \cdot 10^{e-k} + \ldots\,]\,, \qquad d_1 \neq 0\,.$$

Řekneme, že k–tá dekadická cifra d_k aproximace $\tilde{x}$ je platná, jestliže

$$|\tilde{x} - x| \leq 5 \cdot 10^{e-k}, \tag{1.6}$$

tj. když se $\tilde{x}$ liší od x nejvýše o 5 jednotek řádu příslušného následující cifře. Platí-li nerovnost (1.6) pro $k \leq p$, ale pro $k = p + 1$ už neplatí, říkáme, že $\tilde{x}$ má p platných cifer. Číslo $\tilde{x} = \pm\, d_1 d_2 \ldots d_p \cdot 10^{e+1-p}$, které má všech p cifer platných, je správně zaokrouhlenou hodnotou čísla x.

Platné dekadické cifry. Nechť $\tilde{x}$ je aproximace čísla x, kterou zapišme v mocninném dekadickém rozvoji jako

$$\tilde{x} = \pm [\, d_1 \cdot 10^e + d_2 \cdot 10^{e-1} + \dots + d_k \cdot 10^{e+1-k} + d_{k+1} \cdot 10^{e-k} + \dots] \,, \qquad d_1 \neq 0\,.$$

Řekneme, že $k-$tá dekadická cifra d_k aproximace $\tilde{x}$ je platná, jestliže

$$|\tilde{x} - x| \leq 5 \cdot 10^{e-k}, \tag{1.6}$$

tj. když se $\tilde{x}$ liší od x nejvýše o 5 jednotek řádu příslušného následující cifře. Platí-li nerovnost (1.6) pro $k \leq p$, ale pro $k = p + 1$ už neplatí, říkáme, že $\tilde{x}$ má p platných cifer. Číslo $\tilde{x} = \pm\, d_1 d_2 \dots d_p \cdot 10^{e+1-p}$, které má všech p cifer platných, je správně zaokrouhlenou hodnotou čísla x.

Platná desetinná místa. Řekneme, že aproximace $\tilde{x}$ čísla x má k-té desetinné místo platné, jestliže

$$|\tilde{x} - x| \leq 5 \cdot 10^{-k-1}, \tag{1.7}$$

tj. když se $\tilde{x}$ liší od x nejvýše o 5 jednotek řádu příslušného následujícímu desetinnému místu. Platí-li nerovnost (1.7) pro $k \leq p$, ale pro $k = p + 1$ už neplatí, říkáme, že $\tilde{x}$ má p platných desetinných míst. Ve správně zaokrouhleném čísle je tedy každé desetinné místo platné.

V následující tabulce uvádíme několik příkladů:

x	$\tilde{x}$	platné cifry	platná desetinná místa
284	290	1	–
$-45{,}8472$	$-45{,}798$	3	1
$100{,}002$	$99{,}9973$	4	2
$99{,}9973$	$100{,}002$	5	2
$-0{,}003728$	$-0{,}0041$	1	3
$1{,}841 \cdot 10^{-6}$	$2{,}5 \cdot 10^{-6}$	0	5

Při odečítání dvou blízkých čísel dochází ke ztrátě platných cifer, jak o tom svědčí

Příklad 1.1. Je-li

$$x = 4{,}998949 \cdot 10^1, \quad \tilde{x} = 4{,}999 \cdot 10^1, \quad |\Delta x| = 5{,}10 \cdot 10^{-4}, \quad \left|\frac{\Delta x}{x}\right| \doteq 1{,}020 \cdot 10^{-5},$$

$$y = 5{,}001848 \cdot 10^1, \quad \tilde{y} = 5{,}002 \cdot 10^1, \quad |\Delta y| = 1{,}52 \cdot 10^{-3}, \quad \left|\frac{\Delta y}{y}\right| \doteq 3{,}039 \cdot 10^{-5},$$

pak pro rozdíly $z = y - x$, $\tilde{z} = \tilde{y} - \tilde{x}$ dostáváme

$$z = 2{,}899 \cdot 10^{-2}, \quad \tilde{z} = 3 \cdot 10^{-2}, \quad |\Delta z| = 1{,}01 \cdot 10^{-3}, \quad \left|\frac{\Delta z}{z}\right| \doteq 3{,}484 \cdot 10^{-2},$$

takže $\tilde{z}$ má jen jednu platnou cifru, zatímco $\tilde{x}$ i $\tilde{y}$ mají čtyři platné cifry. $\square$

Příklad 1.2. Nechť $x = 1{,}3262 \pm 5 \cdot 10^{-5}$, $y = -6{,}5347 \pm 5 \cdot 10^{-5}$, $z = 13{,}235 \pm 5 \cdot 10^{-4}$. Máme určit aproximaci funkční hodnoty $f = xy/z$, absolutní a relativní chybu a počet platných cifer výsledku.
Spočteme $\tilde{f} = \tilde{x}\tilde{y}/\tilde{z} \ = -6{,}548031\ldots \cdot 10^{-1}$ a dále podle (1.1) dostáváme

$$\left|\frac{\Delta f}{\tilde{f}}\right| \lesssim \left[\left|\frac{\tilde{y}}{\tilde{z}}\,\Delta x\right| + \left|\frac{\tilde{x}}{\tilde{z}}\,\Delta y\right| + \left|\frac{\tilde{x}\tilde{y}}{\tilde{z}^2}\,\Delta z\right|\right] \left|\frac{\tilde{x}\tilde{y}}{\tilde{z}}\right|^{-1} = \left|\frac{\Delta x}{\tilde{x}}\right| + \left|\frac{\Delta y}{\tilde{y}}\right| + \left|\frac{\Delta z}{\tilde{z}}\right| \doteq 8{,}31 \cdot 10^{-5}$$

Odtud $|\Delta f| \doteq 8{,}31 \cdot 10^{-5} \cdot |\tilde{f}| \doteq 5{,}44 \cdot 10^{-5} < 5 \cdot 10^{-1-3}$, takže (se třemi platnými ciframi) $f = -0{,}6548 \pm 0{,}0001$. $\square$

Obsah

Reálná čísla jsou v počítačích reprezentována v systému čísel s pohyblivou řádovou čárkou (v angličtině floating point numbers). Základní myšlenka je podobná semilogaritmickému zápisu (v angličtině scientific notation), v němž např. číslo 245700 píšeme jako $2{,}457 \cdot 10^5$ a číslo 0,0005768 jako $5{,}768 \cdot 10^{-4}$. V tomto formátu se desetinná čárka pohybuje (v doslovném překladu „plave“) v závislosti na dekadickém exponentu. Formálně lze systém $\mathbb{F}$ normalizovaných čísel pohyblivé řádové čarky charakterizovat čtyřmi celými čísly:

β	základ číselné soustavy ($\beta \geq 2$),
p	přesnost ($p \geq 1$),
$[L\,, U]$	rozsah exponentu ($L < 0 < U$).

Každé číslo $x \in \mathbb{F}$ má tvar

$$x = \pm m \cdot \beta^e\,, \quad \text{kde} \quad m = d_1 + \frac{d_2}{\beta} + \frac{d_3}{\beta^2} + \cdots + \frac{d_p}{\beta^{p-1}}$$

je normalizovaná mantisa, $d_i \in \{0, 1, \ldots, \beta - 1\}$, $i = 1, 2, \ldots, p$, jsou cifry mantisy, p je počet cifer mantisy a $e \in \langle L, U \rangle$ je celočíselný exponent. Normalizace mantisy znamená, že pro $x \neq 0$ je prvá cifra mantisy nenulová, tj. platí $d_1 \geq 1$, takže $1 \leq m < \beta$. Když $x = 0$, pak je nulová mantisa i exponent, tj. $m = e = 0$.

Většina počítačů používá binární aritmetiku, kdy $\beta = 2$. Pro stručnější zápis binárních čísel se běžně používá hexadecimální soustava, v níž $\beta = 16$ (cifry 10 až 15 zapisujeme pomocí písmen A,B,C,D,E,F), a někdy rovněž oktalová soustava, kdy $\beta = 8$. Výsledky výpočtů se zpravidla uvádějí v běžné dekadické soustavě, tj. pro $\beta = 10$.
Množina $\mathbb{F}$ čísel pohyblivé řádové čárky je konečná, počet čísel v ní je

$$2(\beta - 1)\beta^{p-1}(U - L + 1) + 1\,,$$

neboť můžeme volit dvě znaménka, $\beta - 1$ možností pro první cifru mantisy, β možností pro zbývajících $p - 1$ cifer mantisy a $U - L + 1$ možných hodnot exponentu. Poslední jednička odpovídá číslu nula.

Nejmenší kladné číslo v $\mathbb{F}$ je číslo UFL $= \beta^L$ (podle anglického UnderFlow Level), které má první cifru mantisy rovnu jedné, zbývající cifry mantisy nulové a exponent nejmenší možný. Největší číslo v $\mathbb{F}$ je číslo OFL $= (\beta - \beta^{1-p})\beta^U$ (podle anglického OverFlow Level), které má všechny cifry mantisy rovné $\beta - 1$ a exponent největší možný.

Zaokrouhlování. Reálná čísla, která jsou přesně zobrazitelná v systému $\mathbb{F}$, se nazývají strojová čísla. Pokud dané reálné číslo $x \notin \mathbb{F}$, musíme ho aproximovat „blízkým" strojovým číslem, které značíme $\mathrm{fl}(x)$ (podle anglického floating). Standardní způsob je zaokrouhlení: $\mathrm{fl}(x)$ je strojové číslo nejbližší k x (když máme na výběr ze dvou možností, pak vybereme to strojové číslo, které má poslední cifru sudou).

Strojová přesnost. Číslo $\varepsilon_m = \beta^{1-p}$ se nazývá strojové epsilon nebo také strojová přesnost (anglicky „machine epsilon“, označované jako ε_{mach}). Význam čísla ε_m dokládá několik jeho charakteristických vlastností:

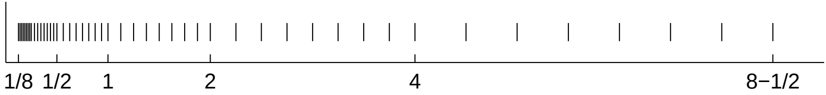
a) v intervalu $\langle \beta^e, \beta^{e+1} \rangle$ jsou strojová čísla rozmístěna rovnoměrně s krokem $\varepsilon_m \beta^e$;

b) největší možná relativní chyba, která vznikne při aproximaci reálného čísla v systému $\mathbb{F}$ pohyblivé řádové čárky, nepřesáhne $\frac{1}{2}\varepsilon_m$, tj. platí $|\mathrm{fl}(x) - x| \leq \frac{1}{2}\varepsilon_m |x|$;

c) ε_m je největší z kladných čísel ε, pro která $\mathrm{fl}(1 + \frac{1}{2}\varepsilon) = 1$.

Je dobré uvědomit si, že $\varepsilon_m \in \mathbb{F}$, jen když $1 - p \geq L$.

Příklad 1.3. Prozkoumejme, jaká čísla můžeme zobrazit v modelovém binárním systému $\mathbb{F}$ v případě, že mantisa má $p = 4$ cifry a exponent e je omezen zdola číslem $L = -3$ a shora číslem $U = 2$, tj. $-3 \leq e \leq 2$.
Umístění kladných strojových čísel na číselné ose je patrné z obrázku 1.1: nejmenší z nich UFL $= 2^{-3} = 1/8$ a největší



Obr. 1.1: Strojová čísla

OFL$= (2 - 2^{-3}) \cdot 2^2 = 8 - 1/2$. Všimněte si, že v každém binárním intervalu $2^e \leq x \leq 2^{e+1}$ jsou čísla rozložena rovnoměrně s krokem $\varepsilon_m 2^e$. Tak třeba mezi 1 a 2, tj. pro $e = 0$, je vzdálenost dvou sousedních čísel rovna $\varepsilon_m = 1/8$. Systém $\mathbb{F}$ obsahuje 48 kladných čísel, 48 záporných čísel a nulu, tj. celkem 97 čísel. $\square$

Standard IEEE. V počítačích vyvinutých po roce 1985 se reálná čísla zobrazují prakticky výhradně podle standardu IEEE, a to zpravidla v těchto přesnostech:

a) Jednoduchá přesnost. Použijí se 4 bajty, tj. 32 bitů, z toho 23 bitů pro mantisu, 8 bitů pro exponent a 1 bit pro znaménko mantisy. Protože mantisa je normalizovaná, pro $x \neq 0$ je $d_1 = 1$. Tato cifra se neukládá, takže počet cifer mantisy $p = 24$. Rozsah exponentu je $-126 \leq e \leq 127$. Zobrazit lze dekadická čísla s absolutní hodnotou v rozsahu

$$\mathrm{UFL} = 2^{-126} \doteq 1{,}2 \times 10^{-38} \quad \text{až} \quad \mathrm{OFL} = (2 - 2^{-23}) \times 2^{127} \doteq 3{,}4 \times 10^{38}$$

a nulu (má všechny bity nulové). Strojová přesnost $\varepsilon_m = 2^{-23} \doteq 1{,}2 \times 10^{-7}$. Říkáme, že mantisa má zhruba 7 dekadických cifer přesnosti.

b) Dvojnásobná přesnost. Použije se 8 bajtů, tj. 64 bitů, z toho 52 bitů pro mantisu, 11 bitů pro exponent a 1 bit pro znaménko mantisy. První bit mantisy se neukládá (pro $x \neq 0$ je $d_1 = 1$), takže mantisa má $p = 53$ cifer. Rozsah exponentu je
$-1022 \leq e \leq 1023$. Zobrazit lze dekadická čísla s absolutní hodnotou v rozsahu

$$\mathrm{UFL} = 2^{-1022} \doteq 2{,}2 \times 10^{-308} \text{ až } \mathrm{OFL} = (2 - 2^{-52}) \times 2^{1023} \doteq 1{,}8 \times 10^{308}$$

a nulu (má všechny bity nulové). Strojová přesnost $\varepsilon_m = 2^{-52} \doteq 2{,}2 \times 10^{-16}$. Říkáme, že mantisa má zhruba 16 dekadických cifer přesnosti.

Podle IEEE standardu existuje také binární reprezentace INF pro výrazy typu $+\infty$ (třeba výsledek operace $1/0$), $-$INF pro výrazy typu $-\infty$ (třeba výsledek operace $-1/0$) a NAN (not a number) pro výrazy typu $0/0$, $\infty - \infty$ a $0 \times (\pm\infty)$ (třeba výsledek operace $1/0 - 2/0$). Systém $\mathbb{F}$ je na většině počítačů rozšířen o tzv. subnormální čísla, což jsou nenulová nenormalizovaná čísla s nejmenším možným exponentem $e = L$. Nejmenší kladné subnormální číslo $\mathrm{UFL}_s = \varepsilon_m \cdot \mathrm{UFL}$, tj. v jednoduché přesnosti $\mathrm{UFL}_s \doteq 1,4 \cdot 10^{-45}$ a ve dvojnásobné přesnosti $\mathrm{UFL}_s \doteq 4,9 \cdot 10^{-324}$.

Počítačová aritmetika. Nechť $x, y \in \mathbb{F}$ jsou strojová čísla, op je některá ze základních aritmetických operací $+, -, \times, /$ a `flop` je odpovídající operace prováděná počítačem v režimu pohyblivé řádové čárky podle IEEE standardu. Pokud je x `flop` $y \in \mathbb{F}$ zobrazitelné strojové číslo, pak x `flop` $y = \mathrm{fl}(x \text{ op } y)$. To znamená, že výsledek aritmetické operace provedené v počítači je stejný, jako když operaci provedeme přesně a pak získaný výsledek vložíme do počítače.

Přetečení, podtečení. Jestliže je absolutní hodnota výsledku aritmetické operace větší než OFL, dochází k tzv. přetečení, a je-li naopak absolutní hodnota nenulového výsledku menší než UFL (resp. UFL_s na počítačích se subnormálními čísly), dochází k tzv. podtečení. Dojde-li při běhu programu k přetečení, systém vydá varování a výpočet přeruší. V případě podtečení situace není tak vážná: výsledek se nahradí nulou a výpočet pokračuje bez přerušení. Reakci programu na přetečení však může poučený programátor sám řídit.

Obsah

Korektní úlohy. Matematickou úlohu lze chápat jako zobrazení $y = f(x)$, které ke každému vstupnímu údaji x z množiny D vstupních dat přiřadí výsledek y z množiny R výstupních dat. Řekneme, že matematická úloha

$$y = f(x)\,, \qquad x \in D\,, \quad y \in R\,,$$

je korektní, když

1) ke každému vstupu $x \in D$ existuje jediné řešení $y \in R$,
2) toto řešení závisí spojitě na vstupních datech, tj. když $x \to a$, potom $f(x) \to f(a)$.

Velkou třídu nekorektních úloh tvoří nejednoznačně řešitelné úlohy. Nekorektní úlohy se nedají rozumně řešit a proto se jimi nebudeme vůbec zabývat.

Podmíněnost úloh. Budeme říkat, že korektní úloha je dobře podmíněná, jestliže malá změna ve vstupních datech vyvolá malou změnu řešení. Je-li $y + \Delta y$ resp. y řešení úlohy odpovídající vstupním datům $x + \Delta x$ resp. x, potom číslo

$$C_p = \frac{|\Delta y|/|y|}{|\Delta x|/|x|} = \frac{|\text{relativní chyba na výstupu}|}{|\text{relativní chyba na vstupu}|} \tag{1.8}$$

(kde místo absolutních hodnot jsou obecně normy, viz kap. 2) nazýváme číslo podmíněnosti úlohy $y = f(x)$. Je-li $C_p \approx 1$, je úloha dobře podmíněná. Pro velká C_p, např. pro $C_p > 100$, je úloha špatně podmíněná. Místo o dobré nebo špatné podmíněnosti mluvíme někdy o malé nebo velké citlivosti vzhledem ke vstupním datům.

Příklad 1.4. Odhadněte číslo podmíněnosti úlohy: stanovit funkční hodnotu (diferencovatelné) funkce $y = f(x)$. Z (1.2) plyne, že

$$C_p \doteq \left| \frac{xf'(x)}{f(x)} \right| . \tag{1.9}$$

Konkrétně pro funkci $f(x) = \operatorname{tg} x$ dostaneme $C_p \doteq |2x/\sin 2x|$. Výpočet $\operatorname{tg} x$ je velmi citlivý pro x blízké celočíselnému násobku $\pi/2$. Třeba pro $x = 1{,}57079$ je $C_p \doteq 2{,}48 \cdot 10^5$.
Výsledek můžeme ověřit také přímým výpočtem podle vzorce (1.8), pro $\Delta x = 10^{-9}$ dostaneme opět $C_p \doteq 2{,}48 \cdot 10^5$. □

Číslo podmíněnosti definované podle (1.8) se někdy označuje jako relativní číslo podmíněnosti. Většinou je to vhodné měřítko citlivosti, je-li však x nebo y rovno nule, použít ho nelze. V takových případech lze zkusit absolutní číslo podmíněnosti definované jako $\bar{C}_p = |\Delta y| / |\Delta x|$.

Příklad 1.5. Posoudíme citlivost výpočtu funkční hodnoty $f(x) = x^2 - 1$. Pro kořeny $x_{1,2} = \pm 1$ není relativní číslo podmíněnosti definováno. Stejný závěr potvrzuje vyjádření $C_p \doteq |2x^2/(x^2 - 1)|$ odvozené podle (1.9). Absolutní číslo podmíněnosti je

$$\bar{C}_p = \left| \frac{f(x + \Delta x) - f(x)}{\Delta x} \right| \doteq |f'(x)| \,,$$

tj. pro $f(x) = x^2 - 1$ je $\bar{C}_p \doteq |2x|$ a speciálně pro $x = \pm 1$ dostaneme $\bar{C}_p \doteq 2$.
□

Stabilita algoritmu. Při realizaci numerické metody na počítači vznikají zaokrouhlovací chyby, nejdříve ve vstupních datech a pak v průběhu výpočtu při provádění aritmetických operací. Chceme-li se při výpočtech vyvarovat nesmyslných výsledků, musíme si vybírat tzv. stabilní algoritmy, které jsou k šíření zaokrouhlovacích chyb málo citlivé. Aby byl algoritmus stabilní, musí být

1) dobře podmíněný, tj. málo citlivý na poruchy ve vstupních datech,
2) numericky stabilní, tj. málo citlivý na vliv zaokrouhlovacích chyb vznikajících během výpočtu.

Příklad 1.6. Kvadratická rovnice $x^2 - 2bx + c = 0$ má řešení

$$x_{1,2} = b \pm d\,, \tag{A_1}$$

kde $d = \sqrt{b^2 - c}$. Jestliže $|b| \doteq |d|$, pak se při výpočtu jednoho z kořenů budou odečítat dvě přibližně stejně velká čísla téhož znaménka, což vede, jak už víme, ke vzniku velké relativní chyby (viz (1.3) a příklad 1.1). Výpočet podle algoritmu (A_1) tedy obecně stabilní není. Existuje však snadná pomoc: protože $x_1 x_2 = c$, můžeme postupovat takto:

$$x_1 = \begin{cases} b + d\,, & \text{je-li } b \geq 0, \\ b - d\,, & \text{je-li } b < 0, \end{cases} \qquad x_2 = c/x_1\,. \tag{A_2}$$

Algoritmus (A_2) nedostatek algoritmu (A_1) odstraňuje, tj. je stabilní. $\square$

Příklad 1.7. Počítejme integrál

$$E_n = \int_0^1 x^n e^{x-1}\,\mathrm{d}x \qquad \text{pro } n = 1, 2, \ldots$$

Integrací per-partes dostaneme

$$\int_0^1 x^n e^{x-1}\,\mathrm{d}x = [x^n e^{x-1}]_0^1 - \int_0^1 n x^{n-1} e^{x-1}\,\mathrm{d}x\,,$$

nebo-li

$$E_n = 1 - nE_{n-1}\,. \tag{F}$$

Protože $E_1 = 1/e$, můžeme při výpočtu E_n postupovat podle algoritmu

$$E_1 = 1/e\,, \qquad E_n = 1 - nE_{n-1}\,, \qquad n = 2, 3, \ldots \tag{A_1}$$

Výpočet jsme provedli na počítači v jednoduché přesnosti (tj. cca na 7 platných cifer), a pro $n = 12$ jsme dostali $E_{12} \doteq -4{,}31$. To je ale zcela nepřijatelný výsledek, neboť pro kladný integrand nemůžeme dostat zápornou hodnotu integrálu! Tento jev je způsoben tím, že při výpočtu E_n se chyba obsažená v E_{n-1} násobí $n-$krát, takže celková chyba roste podobně jako $n!$.

Algoritmus(A_1) je tedy nestabilní. Vzniká otázka, zda můžeme vypočítat E_{12} užitím rekurentní formule (F) tak, aby výsledek měl všech 7 cifer platných. Možné to je, musíme ale použít jiný algoritmus. Přepíšeme-li formuli (F) na tvar

$$E_{n-1} = \frac{1 - E_n}{n}\,,$$

bude se chyba vstupující do každého kroku dělit n. Z odhadu

$$E_n = \int_0^1 x^n e^{x-1}\,\mathrm{d}x \le \int_0^1 x^n\,\mathrm{d}x = \frac{1}{n+1}$$

vyplývá, že $E_n \to 0$ pro $n \to \infty$. Při výpočtu proto budeme postupovat podle algoritmu

$$E_N = 0\,, \qquad E_{n-1} = \frac{1 - E_n}{n}\,, \qquad n = N, N-1, \dots\,, \tag{A_2}$$

kde N je dostatečně velké. Zvolíme-li $N = 20$, dostaneme $E_{12} \doteq 7{,}177325 \cdot 10^{-2}$, což je hodnota, která má 7 cifer platných.

Jestliže výpočet provádíme ve dvojnásobné přesnosti (cca 16 platných cifer), dostaneme obdobné výsledky. Rozdíl je pouze v tom, že výpočet algoritmem (A_1) se „pokazí“ až pro větší n; pro $n = 20$ už ale vyjde nesmyslná hodnota $E_{20} \doteq -30{,}19$. Stabilním algoritmem (A_2) pro $N = 35$ dostaneme $E_{20} \doteq 4{,}554488407581805 \cdot 10^{-2}$ s 16-ti platnými ciframi. □

Další příklady věnované podmíněnosti problémů a zejména algoritmů uvedeme postupně v následujících kapitolách.