ZÁLUDNOSTI V PROGRAMOVÁNÍ VĚDECKO-TECHNICKÝCH VÝPOČTŮ

RNDr. Jiří Hřebíček, CSc., RNDr. Ivan Kopeček, CSc.

Úvod

V oblasti vědeckotechnických výpočtů (dále jen VTV) číhá na programátora i výzkumného pracovníka řada úskalí, specifických pro tuto oblast. I velmi zkušení programátoři VTV se mnohdy znovu přesvědčují o tom, jak mohou být tato úskalí záludná a jak se dokážou objevovat v nových nečekaných situacích. Stejně tak i výzkumní pracovníci používající kalkulaček a programovatelných kalkulátorů jsou mnohdy překvapeni, když narazí na tato úskalí a i při obecně uznávaném a známém algoritmu dostanou nesprávné výsledky. Tento příspěvek se stručně zmíní o některých "záludnostech" se kterými je možno se v oblasti VTV setkat.

1. Zaokrouhlovací chyby

VTV pro fyzikální, technické, ekonomické a další vědní obory se vyznačují společným rysem. Je to využívání matematických pojmů a numerických metod při formulaci problému, který je výzkumným pracovníkem a pak matematikem-analytikem postaven před programátora. Původní fyzikální či technický problém je zde reformulován v jazyce matematické analýzy, algebry, numerické matematiky, teorie grafů a pod. Často je ovšem na programátorovi - matematikovi, aby do tohoto jazyka původní problém převedl. V každém případě je nakonec vlastní programátorskou prací ve VTV převedení matematické formulace problému na matematickou formulaci řešení a posléze na algoritmický zápis řešení v programovacím jazyce na daném počítači.

O úskalích, která zde při aplikaci matematiky a vytvoření algoritmu a jeho naprogramování bylo již pojednáno v /1 / na Programování '80 a proto se zde nebudeme touto zajímavou a někdy přímo pro černý humor stvořenou problematikou zabývat.

Převedení matematických veličin a vztahů do programovacího jazyka má některé aspekty, které jsou z našeho hlediska důležité. Především nekonečná neomezená množina reálných čísel se transformuje na konečnou a omezenou množinu hodnot typu REAL. Dále při provádění aritmetických operací dochází jak v počítači, tak kalkulačce či kalkulátoru k zaokrouhlovacím chybám. V důsledku toho řada matemati-

kých tvrzení ztrácí v "počítačovém prostředí" svou platnost a bezprostřední použití některých vět, vzorců a metod může vést k velmi špatným výsledkům.

Uveďme několik příkladů:

1.1. Každý absolvent VŠ a dnes i středoškolák zná, že derivace funkce $f(x)$ v bodě A je definována vztahem

$$f'(A) = \lim_{\Delta x \to 0} \frac{f(A + \Delta x) - f(A)}{\Delta x}$$

Tento vztah (zhruba řečeno) říká, že aproximace hodnoty $f'(A)$ hodnotou $(1/\Delta x)(f(A + \Delta x) - f(A))$ bude tím lepší, čím menší Δx použijeme pro výpočet. V "počítačovém prostředí" se v důsledku zaokrouhlovacích chyb a ztráty platných cifer při odečítání blízkých čísel ukazuje, že použití tohoto diferenčního vzorce vede pro velmi malá Δx ke katastrofálně špatným výsledkům. Ještě horší je situace při aproximování vyšších derivací diferenčními vzorci.

1.2. K čemu může odečítání velmi blízkých malých čísel vést, ukazuje užití Taylorova rozvoje funkce e^x v bodě 0

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

při výpočtu e^x. Pomocí tohoto vztahu na obyčejné kalkulačce o pětimístné mantise dostaneme

$$e^{-5,5} = 0,0026363.$$

Výpočet jsme ukončili, když další sumační členy neměly vliv na výsledek, což nastane po 25tí členech alternující řady

$$e^{-5,5} \approx 1,0000 - 5,5000 + 15,125 - 27,730 + \dots.$$

Protože přesný výsledek je $e^{-5,5} = 0,0040877$, je chyba plných 36 %!

1.3. Pro snažší ilustraci řešme na fiktivním dekadickém počítači s tříměstnou mantisou soustavu lineárních rovnic

$$0,0001x + y = 1$$
$$x + y = 2$$

Eliminováním x z druhé rovnice dostaneme na počítači

$$0,000100x + 1,00y = 1,00$$
$$- 10000y = - 10000.$$

Odtud zpětným chodem dostaneme $x = 0, y = 1$. Zaokrouhlené přesné řešení je přitom

$$x = 1,00, \quad y = 0,999.$$

Zcela špatný výsledek x vzniknul kombinací zaokrouhlovacích chyb s nestabilním numerickým postupem. Pivotováním matice soustavy rovnic tento nežádoucí efekt zcela odstraníme, viz /6/.

1.4. Jak známo platí, že

$$e = \lim_{N \to \infty} (1 + 1/N)^N \approx N$$

Následující tabulka, která uvádí hodnoty aproximace čísla e podle uvedeného vztahu pro N = 10, 11, ... při přesnosti na 16 dekadických míst, ilustruje vliv zaokrouhlovacích chyb na numerický výpočet.

N	aproximace e
10	2,71828
11	2,71822
12	2,71792
13	2,71611
14	2,71611
15	2,43070
16	1,0
17	1,0
	atd.

(2) Formulace problému

V některých případech může potíže způsobit už sama matematická formulace problému. Naopak, vhodná formulace může často odstranit potíže, které by vznikly při rutinním postupu. K stručné ilustraci nám poslouží tyto příklady, další budou uvedeny při prezentaci příspěvku.

2.1. Vraťme se k příkladu 1.2. Výpočet $e^{-5,5}$ můžeme reformulovat jako výpočet převrácené hodnoty k $e^{5,5}$. Při použití Taylorova rozvoje k výpočtu $e^{5,5}$ nevzniknou problémy, protože všechny členy rozvoje budou kladné. Výpočtem na našem počítači dostaneme výsledek $e^{-5,5} = 1/e^{5,5} = 0,0040865$ s přesností na 0,007 %.

2.2. Máme vypočítat integrál

$$J = \int_{0,01}^{1,00} \frac{dx}{e^x - 1} = 4,15149087$$

Při použití notoricky známé Simpsonovy metody s dělením intervalu na 2^n podintervalů bude zapotřebí k dosažení přesnosti na 9 desetinných

míst 4096 uzlů. Pomalá konvergence Simpsonovy metody je způsobena tím, že paraboly, kterými se v této metodě aproximuje integrand, nevystihnou v okolí bodu 0,01 dostatečně chování integrandu, který pro $x \to 0$ konverguje k ∞. Jestliže si však uvědomíme, že integrand se chová v okolí 0,01 jako $1/x$, potom po přeformulování

$$J = \int_{0,01}^{1,00} \left(\frac{1}{e^x - 1} - \frac{1}{x} \right) dx + \int_{0,01}^{1,00} \frac{dx}{x}$$

je hodnota integrandu prvního integrálu v $x = 0$ rovna $-0,5$. Na dosažení stejné přesnosti při integraci prvního integrálu Simpsonovou metodou stačí nyní rozdělení intervalu na pouze 16 podintervalů. Druhý integrál lze spočítat analyticky. Uvedená reformulace vede na 128-násobné zrychlení výpočtu.

3. Stabilita problému a algoritmu

Další úskalí, která čekají programátora mohou být způsobena následujícími důvody:

a) Problém je citlivý na změnu vstupních dat. Malá změna počátečních dat může způsobit velkou změnu výsledku.

b) Použitý algoritmus je citlivý na zaokrouhlovací chyby. Jejich výskyt může způsobit velkou chybu výsledku.

V prvním případě mluvíme o nestabilitě problému, v druhém případě o nestabilitě algoritmu. V analogickém významu ke stabilitě (nestabilitě) se někdy používá pojmu dobré (špatné) podmíněnosti.

3.1. Příkladem nestabilního problému je řešení systému lineárních rovnic $Ax = b$, kde $b = (1,1)^T$ a

$$A = \begin{pmatrix} 1 & 1 \\ 1 & 1,000001 \end{pmatrix}.$$

Změna prvku a_{22} o 0,000001 totiž způsobí maximální možnou "změnu" výsledku, změní totiž regulární matici A na singulární matici

$$A' = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}.$$

3.2. Pro mnohé je překvapující, že nalezení menšího z dvojice reálných kořenů kvadratické rovnice

$$x^2 - 2a_1 x + a_2 = 0$$

pro $a_1 >> 1$, $0 < a_2 << 1$, daného známým vzorcem

$$x_2 = a_1 - \sqrt{a_1^2 - a_2}$$

vede v důsledku odečítání blízkých hodnot a_1 a $\sqrt{a_1^2 - a_2}$ na nestabilní algoritmus. Zapíšeme-li však x_2 ve tvaru

$$x_2 = \frac{a_2}{a_1 + \sqrt{a_1^2 - a_2}}$$

je odpovídající algoritmus stabilní.

Otázka stability numerických problémů a algoritmů je jedna z nejdůležitějších v oblasti VTV. Mnohdy se za dosažení stability algoritmu platí snížením jeho rychlosti, jak tomu např. často bývá při řešení časových problémů parciálních diferenciálních rovnic. Tento významný a složitý problém přesahuje možnosti tohoto příspěvku a proto se zde odkážeme např. na /9 /.

4. Numerický výpočet a specifické "počítačové prostředí"

Numerický výpočet bývá někdy ovlivněn "počítačovým prostředím". "Počítačovým prostředím" zde rozumíme hardware a software použitého výpočetního systému. Zvláště specifika operačního systému, překladačů, podpůrných programových prostředků a počítačových konstant může podstatně ovlivňovat výpočet. Např. při použití různých typů překladačů (např. FORTRAN-G, FORTRAN-H) můžeme dojít k různým výsledkům v důsledku numerické nestability kombinované s tím, že oba překladače přeloží zdrojový tvar sice ekvivalentně, co se týče algebraické reprezentace, ale jinak, co se týče strojových instrukcí. V důsledku zaokrouhlovacích chyb totiž v počítači neplatí základní algebraické zákony (např. distributivní zákon atd.).

Problémem při numerických výpočtech ve VTV bývají rovněž různě naprogramované vnitřní funkce. Uveďme zkušenosti z implementace knihovny NAG /3 / na počítači Prime 400, kde zhavaroval podprogram pro výpočet vlastních vektorů vlivem nepřesností při výpočtu funkce DSQRT. Na počítači Honeywell 66 opět zhavaroval podprogram, který předpokládal, že $|\cos(x)| \leq 1$, což porušila funkce DCOS. Proto bývá vhodné zavést parametry závislé na "prostředí" (definující různé hardwarově a softwarově závislé konstanty), viz /2 /.

5. Volba numerické metody a diskretizace

Volba numerické metody může hrát důležitou roli pro efektivnost výpočtu. Se zlou by se potázal programátor, který by se rozhodl řešit větší systém lineárních rovnic $Ax = b$ Cramerovým pravidlem, které vyžaduje $O(n!)$ aritmetických operací (kde n je hodnost matice). Už pro nepříliš velká n (např. pro $n = 20$) je výpočet s časových důvodů nemožný. Naproti tomu Gaussova eliminační metoda potřebuje zhruba $\frac{n^3}{3}$ operací, takže pro $n = 20$ u většiny počítačů trvá výpočet méně než 0,1 sec. Naproti tomu při řešení velkých soustav lineárních rovnic s řídkou pásovou maticí soustavy může být výhodnější použití iteračních metod, (které jsou pomalejší pro plné matice), které díky řídkosti matice soustavy mohou dát dostatečně přesné výsledky v kratším čase, než Gaussova eliminační metoda.

V řadě případů se programátor musí rozhodnout pro určitou diskretizaci problému. Např. při řešení okrajového problému

$$\varepsilon y'' + y' = 1 \qquad y(0) = 0, \quad y(1) = 0$$

metodou konečných prvků je třeba se rozhodnout na jaký počet subintervalů se rozdělí interval $\langle 0,1 \rangle$. Když ε je dostatečně velké, např. $\varepsilon = 1$, stačí na dosažení dostatečné přesnosti dělení na 10-ti subintervalů. Toto dělení však dá pro $\varepsilon = 0,01$ katastrofální výsledky. Když se však použije vhodného nerovnoměrného dělení, vystačíme i zde s 10 - 20 dílky, viz /6/. Vodítkem zde může být např. predikce charakteru výsledků z hlediska fyzikální (či jiné) interpretace v souvislosti s dobrou znalostí používané metody.

6. Závěr

Náš příspěvek zdaleka nevyčerpává všechny charakteristické situace, kdy se programátor VTV střetává s úskalími, vyplývajícími ze specifik VTV. Navíc jsme se zde zaměřili výhradně na numerické otázky výpočtů a nechali jsme stranou tu část VTV, která má odlišný, převážně diskrétní charakter (např. problémy vedoucí na aplikaci teorie grafů a pod.). Pro detailnější studium otázek, kterými jsme se zde zabývali, se odkazujeme na seznam literatury (odkud byly také převzaty některé z uvedených příkladů).

Literatura

/1/ Bébr,R.: Zajímavosti ze světa vědeckotechnických výpočtů, Sborník Programování '80, Havířov 1980.

/2/ Du Croz,J.: Programming Languages for Numerical Subroutine Libraries, NAg Newsletter 1/82, 1982.

/3/ Hague,S.J., Ford,B.: Tools for Numerical Software Engeneering, NAg Newsletter 1/81, 1981.

/4/ Henzl,P., Hřebíček,J.: Programování vědeckotechnických výpočtů, Sborník Programování `81, Havířov 1981.

/5/ Hřebíček,J., Kopeček,I.: Tvorba, ladění a testování numerického software, Sborník Programování `83, Ostrava 1983.

/6/ Kopeček,I.: Algoritmy řešení problému konvekce a difuse s malým koeficientem difuse metodou konečných prvků, Sborník Algoritmy `83, Vysoké Tatry 1983.

/7/ Mikloško,J.: "Patologické" javy v numerickej matematike, Sborník Software a algoritmy numerické matematiky, Nové Město na Mor., 1979.

/8/ Ralston,A.: Základy numerické matematiky, Academia, Praha 1973.

/9/ Lambert,J.D.: Computational Methods in Ordinary Differential Equations, J. Wiley, London 1973 .