Poznámky k některým tématům z přednášky TMF057 - Numerické metody pro teoretické fyziky.

Martin Čížek

14. ledna 2014

Obsah

Úvod k této verzi

Tento text představuje poznámky k přednášce numerických metod pro teoretické fyziky z roku 2013. Cílem přednášky není provést úplný výčet numerických metod a jejich důkladnou analýzu. Snažil jsem se spíše vybrat nejdůležitější témata numerické matematiky a předvést posluchačům principy a úskalí nejčastěji používaných metod a dále předvést způsob myšlení, který vede k odvození a analýze některých postupů. Doufám, že to posluchačům usnadní další studium a především psaní vlastních programů. V průběhu výkladu se rovněž snažím upozornit na literaturu snadno dostupnou v češtině a rovněž na anglicky psanou literaturu, která většinou patří do jedné z těchto dvou kategorií. Buď jde o knihy přehledně shrnující praktické stránky použití numerických metod, nebo jde o literaturu která umí zprostředkovat radost ze souvislostí a porozumění některým aspektům numerických metod.

Za základní literaturu pro praktické použití metod považuji Numerické recepty [1] a každému posluchači, chystajícímu se řešit nějaký konkrétní problém doporučuji nejdříve konzultovat tento zdroj.

Jde o předběžnou verzi textu, která má usnadnit přípravu na zkoušku v ZS 2013-14. Některé kapitoly jsem vůbec nestihl napsat a uvádím jen doporučenou studijní literaturu. Omluvte prosím veškeré nedostatky. Případné připomínky k textu vítám.

Kapitola 1

Reprezentace čísel, zaokrouhlovací chyba.

Cílem této kapitoly je seznámit se s některými specifiky počítání v pohyblivé řádové čárce. Seznámíme se s pojmy zaokrouhlovací chyba a její nadměrné narůstání v některých příkladech ("cancelation", "smearing"). Dále se stručně seznámíme s problematikou stability algoritmu a rovněž si připomeneme pojem podmíněnosti úlohy. Všechny tyto pojmy nás budou dále provázet celým tímto textem.

Z přednášek z programování víte, že reálná čísla jsou v programovacích jazycích reprezentována pomocí pohyblivé řádové čárky ve tvaru:

$$\pm m \times B^e, \tag{1.1}$$

kde m je tzv. mantisa, e je exponent a B je základ číselné soustavy v níž pracujeme (většinou 2 nebo 10). Přesná reprezentace není pro náš výklad příliš důležitá, každopádně budeme předpokládat, že každé reálné číslo x je reprezentováno číslem $\bar{x}$, tak že

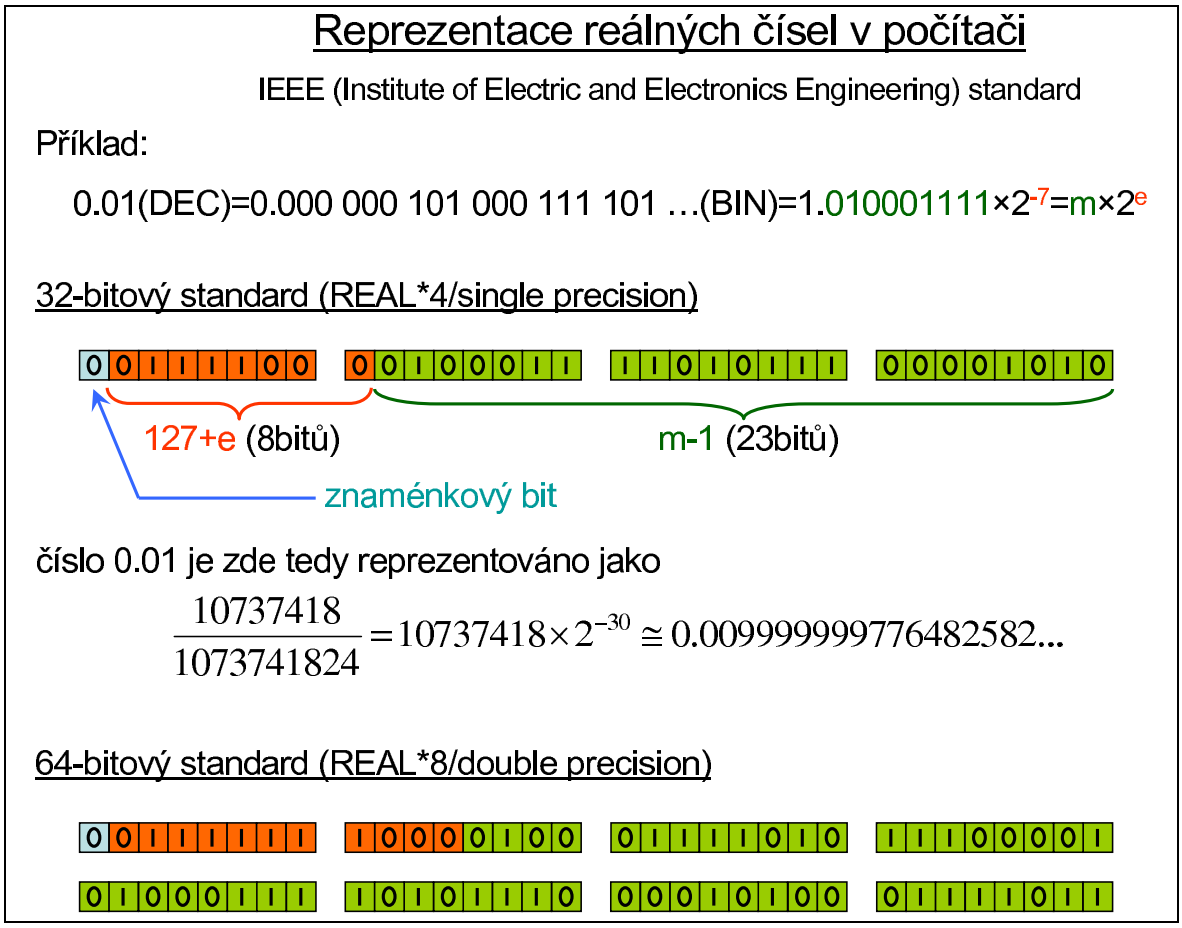
$$\bar{x} = x + \Delta x = x(1 + \epsilon_x), \tag{1.2}$$

kde $|\epsilon_x| < \epsilon$. Číslo ϵ (strojové epsilon), určuje maximální relativní chybu reprezentace reálných čísel. O číslech Δx a ϵ_x budeme mluvit jako o zaokrouhlovací chybě popřípadě relativní zaokrouhlovací chybě. V IEEE standardu pro reprezentaci reálných čísel v pohyblivé řádové čárce je $\epsilon = 2^{-23} \simeq 10^{-7}$ pro reálná čísla v jednoduché přesnosti a $\epsilon = 2^{-52} \simeq 2 \times 10^{-16}$ pro přesnost dvojitou[1].

Při numerických výpočtech pomocí dané reprezentace reálných čísel v počítači je potřeba počítat se zaokrouhlovací chybou, jak při zadání vstupních dat, tak v průběhu veškerých aritmetických operací, z nichž se sestává algoritmus výpočtu hledané veličiny. Numerická matematika se zabývá nejen samotným návrhem a konvergencí algoritmů pro řešení matematických úloh pomocí počítače, ale podstatnou její část tvoří právě analýza vlivu zaokrouhlovacích chyb na správnost výpočtu.

Je potřeba si uvědomit, že některé axiomy reálné aritmetiky nejsou v její počítačové reprezentaci splněny. Například sčítání několika čísel záleží na pořadí, nebo nemusí být vždy splněny

[1] Někdy se udává strojové epsilon poloviční, což odpovídá tomu, že ke každému x najdeme ne nejbližší nižší, ale nejbližší číslo, které lze reprezentovat v daném standardu zápisu pomocí pohyblivé řádově čárky. Taková nejednoznačnost je pro nás nepodstatná, vzhledem k tomu, že nám jde jen o řádové odhady chyby.



Obrázek 1.1: Standardní reprezentace reálných čísel v počítači.

některé nerovnosti (například nerovnost mezi geometrickým a aritmetickým průměrem). Pěkné pojednání o tomto tématu naleznete na

`http://www.lahey.com/float.htm`

Na první pohled se může zdát, že vzhledem k tomu, jak malá je zaokrohlovací chyba, nebudou rozdíly mezi přesným výsledkem a jeho počítačovou reprezentací podstatné. V této úvodní kapitole si ukážeme několik příkladů vlivu zaokrouhlovací chyby na výsledky jednoduchých výpočtů včetně případů, kdy může zaokrouhlovací chyba vést k velmi nepřesnému či dokonce nesmyslnému výsledku.

Příklad 1: (*Výsledek závisí na pořadí sčítanců.*) Napište krátký program, který vypočte sumu $S_n = 1 + \sum_{k=1}^{n} \frac{1}{k^2+k}$. Výsledek lze spočíst přesně, přičemž dostaneme $S_n = 2 - 1/(n+1)$ (nápověda: rozložte členy sumy do parciálních zlomků). Porovnáním tohoto přesného výsledku a výsledku běhu programu, zjistíte, že zaokrohlovací chyba roste s n přibližně jako $\sqrt{n}\epsilon$. Zkuste napsat program, který vypočte stejnou sumu, ale přičítá sčítance odzadu. Výsledek: zaokrouhlovací chyba je menší než ϵ pro jakkoli velké n.

Příklad 2: (*Odčítání skoro stejně velkých čísel: „cancellation“.*) Při sčítání (odčítání) dvou čísel x_1 a x_2, lze zaokrouhlovací relativní chybu omezit shora výrazem

$$\frac{|\Delta_{x_1}| + |\Delta_{x_2}|}{|x_1 \pm x_2|} + \epsilon.$$

První část výrazu je vliv zaokrouhlovací chyby sčítanců; dodatečné epsilon je kvůli zaokrouhlení výsledku. Vidíme, že relativní chyba výsledku, může být značně velká, pokud odčítáme přibližně stejná čísla. Zkuste si spočíst na počítači v aritmetice s pohyblivou řádovou čárkou výraz $\sqrt{9876}-\sqrt{9875}$ a porovnejte jej s výsledkem vyčíslení matematicky ekvivalentího výrazu $(\sqrt{9876}+\sqrt{9875})^{-1}$. Zatímco v prvním způsobu výpočtu ztrácíme přesnost (asi 3-4 cifry od konce) druhý způsob si zachovává přesnost na úrovni strojového ϵ.

S tímto typem problému se setkáme při **řešení kvadratické rovnice**. Pro některé hodnoty koeficientů můžeme dostat výsledek zatížený značnou chybou pokud nevěnujeme numerickému výpočtu kořenů dostatečnou pozornost. Zkuste si například vypočíst numericky oba kořeny rovnice $x^2-1634x+2=0$. Označme x_1 větší z obou kořenů. Potom x_2 můžete spočíst přímo ze známého vzorce pro kořeny kvadratické rovnice, a nebo využít vztahu $x_2 = 2/x_1$. Při prvním způsobu výpočtu ztrácíte pět desetiných míst přesnosti, zatímco chyba při výpočtu druhým způsobem je v mezích strojového ϵ. Více o správném numerickém výpočtu kořenů kvadratické rovnice si přečtěte v Numerických receptech [1] (kapitola 5.6).

Příklad 3: (*Součet mnoha čísel, jehož výsledek je malý: „smearing“.*) S podobným jevem jako v předchozím příkladě (ale ve skrytější podobě) se setkáme například při vyčíslování řad. Zkuste spočíst hodnotu e^x pomocí Taylorova rozvoje. Pro malé hodnoty x dostáváte velmi přesný výsledek a i pro větší, kladná x, dostanete přesný výsledek pokud vezmete dost členů Taylorova rozvoje. (Dá se výsledek ještě zpřesnit přeuspořádáním členů řady? Napište program, který najde vždy výsledek s relativní chybou srovnatelnou se strojovým ϵ.) Pro větší záporná x tento výpočet selže a dostáváme nesmyslné výsledky. Důvod je ten, že velmi malé číslo e^{-x} se snažíme najít jako součet poměrně velkých členů řady. Zaokrouhlovací chyba těchto členů je vůči malému očekávanému výsledku velká a skutečný numerický součet řady je dominován právě zaokrouhlovacími chybami. Náprava je snadná. Stačí využít relace $e^x = 1/e^{-x}$.

Následující příklad je převzat z [7] a bude nás provázet do konce této kapitoly. Definujme veličinu

$$I_n(a) = \int_0^1 \frac{x^n}{x+a}. \tag{1.3}$$

Pro malá n snadno spočtete I_n analyticky. Vzorec pro obecné n lze nalézt použitím binomického rozvoje na čitatel, pokud napíšete $x = (x+a)-a$, takže po triviální integraci

$$I_n(a) = \sum_{k=0}^{n-1}(-a)^k \binom{n}{k}\{(1+a)^{n-k}-a^{n-k}\}\frac{1}{n-k} + (-a)^n \ln\frac{a+1}{a}. \tag{1.4}$$

Toto je přesná formule. Přesto jejím použitím pro $a = 10$ zjistíte, že pro $n > 10$ začnou vycházet nesmysly (z definice (1.3) je ihned zřejmé $0 < I_n < 1$, a přesto dostanete záporný výsledek). Důvod je třeba hledat ve skládání zaokrouhlovacích chyb. Například pro $n = 10$ má šestý člen sumy (tj. $k = 5$) hodnotu asi 3×10^{11}. Sčítáním takto velkých čísel máte nakonec dostat výsledek jehož velikost je menší než jedna. Jde tedy o jasný příklad „smearingu“ (viz předcházející odstavec). Metodu výpočtu $I_n(a)$ pro libovolnou hodnotu n a a si ukážeme v následujícím odstavci.

1.1 Stabilita algoritmu

V příkladech z předchozí kapitoly jsme viděli, že veličiny dané jakýmsi matematickým výrazem můžeme vypočíst několika způsoby, neboli několika postupy (algoritmy). Přitom ne všechny

způsoby jsou stejně dobré pro použití v aritmetice s pohyblivou řádovou čárkou. S tím souvisí pojem stability algoritmu.

Definice: *Stabilita algoritmu AA.* Řekneme, že určitý postup výpočtu veličiny $f = f(x)$ závisející na vstupních datech x je stabilním algoritmem, pokud provedením tohoto postupu v aritmetice s pohyblivou řádovou čárkou dá pro všechna x výsledek $\tilde{f}$, který se od přesného f liší relativní chybou řádu $O(\epsilon)$, tj. při použití aritmetik s pohyblivou řádovou čárkou se zmenšujícím se strojovým ϵ lze chybu odhadnout shora výrazem[2] $c\epsilon$, kde c je nějaká konstanta nezávisející na x.

V praxi musíme navíc požadovat, aby konstanta c nebyla příliš velká. S konstantou řádu $c = \epsilon^{-1}$ pro strojové ϵ na počítači, který máme k dispozici stejně, nedostaneme rozumný výsledek.

Na předchozí stránce jsme viděli, že výpočet $\sqrt{x+1} - \sqrt{x}$ pro kladné x je třeba provést na základě algoritmu založeném na vzorci $\sqrt{x+1} - \sqrt{x} = (\sqrt{x+1} + \sqrt{x})^{-1}$. Absolutní chybu prvního vzorce lze odhadnout výrazem $2\sqrt{x}\epsilon$, tj. relativní chyba se pro velká x chová jako $2\epsilon x$, a tedy pro velká x roste nade všechny meze. Pro druhý výraz dostaneme stejnou absolutní chybu při výpočtu jmenovatele, tj. relativní chyba jmenovatele je nejvýše 2ϵ. Při výpočtu převrácené hodnoty se relativní chyba nemění tudíž chybu výsledku odhadneme výrazem 2ϵ a algoritmus pomocí tohoto vzorce je tedy stabilní.

Příklad: *Stabilita výpočtu posloupnosti zadané rekurentím vzorcem.* Vraťme se k výpočtu integrálu $I_n(a)$ definovaného v předchozí kapitole. Snadno zjistíte, že

$$I_n = \int_0^1 \frac{x^{n-1}(x+a-a)}{x+a}dx = \int_0^1 x^{n-1} - a\int_0^1 \frac{x^{n-1}}{x+a}dx = \frac{1}{n} - aI_{n-1}. \tag{1.5}$$

To je rekurentní vztah, který umožňuje vypočíst postupně I_1, I_2, …, I_n z hodnoty $I_0 = \ln\frac{1+a}{a}$. Vyzkoušejte si naprogramovat tento postup v aritmetice s pohyblivou řádovou čárkou a použijete jej opět pro $a = 10$. Opět zjistíte, že už pro poměrně malé hodnoty n dostanete záporné (a tudíž nesmyslné) hodnoty I_n. Na vině je opět zaokrouhlovací chyba. Předpokládejme, že do posloupnosti I_n v určitém bodě zaneseme chybu Δ_n, tj. změníme $I_n \to \tilde{I}_n = I_n + \Delta_n$. Snadno zjistíme, že chyba se šíří dále podle vzorce $\Delta_{n+1} = -a\Delta_n$, tj. po k krocích je

$$\Delta_{n+k} = (-a)^k\Delta_n. \tag{1.6}$$

Pro $a = 10$ tudíž chyba vzroste v každém kroku o jeden řád. Ve dvojité přesnosti, se strojovým $\epsilon \simeq 10^{-16}$ převáží zaokrouhlovací chyby po 16 krocích nad správným výsledkem. Vidíme tudíž, že výpočet integrálu $I_n(a)$ pomocí rekurentní formule (1.5) není stabilní (c v definici stability je rovno a^n a nelze jej omezit shora konstantou nezávislou na vstupních datech a, n).

Náprava tohoto problému není obtížná. Pojďme se podívat, jak se chová chyba při použití vzorce (1.5) odzadu:

$$\Delta_{n-k} = (-1/a)^k\Delta_n. \tag{1.7}$$

Například pro $a = 10$ se tedy chyba zmenšuje o jeden řád s každou iterací. Pro $a > 1$ můžeme tedy generovat posloupnost I_n, I_{n-1}, I_{n-2} pomocí relace (1.5). Chyba se každou iterací zmenší faktorem $1/|a|$. Volbou dostatečně velkého n a zahozením prvních pár členů posloupnosti dostaneme tedy správné výsledky i pro úplně špatnou volbu počáteční hodnoty I_n, například $I_n = 0$. Docházíme k zajímavému výsledku, že zatímco přesný výpočet posloupnosti odpředu

[2]Tento požadavek může být pro některé problémy příliš silný. Potom vystačíme s odhadem chyby $O(\epsilon^\alpha)$, kde $0 < \alpha < 1$. Viz například odstavec o numerickém derivování.

ze správné hodnoty I_0 vede k nesprávným číslům, přibližný výpočet posloupnosti odzadu z nesprávné hodnoty I_n vede k přesnému výsledku (s chybou řádu strojového ϵ až na prvních pár členů posloupnosti).

1.2 Podmíněnost úlohy

Tuto úvodní kapitolu zakončíme definicí pojmu podmíněnosti úlohy. Předchozí úvahy a příklady se týkaly řešení nějaké matematické úlohy, nalezení čísla $f = f(x)$ (nebo množiny čísel $f \equiv f_i$) na základě vstupních dat x. Přitom jsme se soustředili na různé algoritmy výpočtu veličiny f a na způsob, jakým malé chyby vstupující v různých místech do algoritmu výpočtu ovlivňují výsledek. Některé úlohy mohou být samy o sobě, bez ohledu na způsob výpočtu, obtížné, neboť výsledek silně závisí na vstupních datech.

Příklad: V knize „To snad nemyslíte vážně“ popisuje Richard Feynman, jak se vsadil, že dokáže vypočíst během minuty výsledek jakékoli početní úlohy s přesností na deset procent. Kolegové mu dávali různé úlohy jako výpočty ošklivě vypadajících integrálů či řešení diferenciálních rovnic. Díky své genialitě a zběhlosti v přibližných výpočtech Feynman vždy vyhrál, až šel kolem jeho kamarád matematik a ten mu řekl, ať spočte $\tan(10^{100})$. Toto je příkladem špatně podmíněné úlohy. Pokud by měl mít Feynman naději spočíst výsledek s relativní přesností 10^{-1}, musel by znát vzdálenost čísla 10^{100} od nejbližšího násobku π s absolutní přesností řádu 10^{-1}, což pro argument velikosti 10^{100} dává relativní přesnost 10^{-101}. Kamarád matematik si prostě uvědomil, že mu musí dát úlohu, v jejímž výsledku se extrémně násobí chyba počátečních dat.

Definice. *Číslem podmíněnosti úlohy* nazveme

$$\hat{\kappa}(x) = \lim_{\epsilon \to 0} \sup_{\|\delta x\| < \epsilon} \frac{\|\delta f\|}{\|\delta x\|} \tag{1.8}$$

(absolutní číslo podmíněnosti), nebo

$$\kappa(x) = \lim_{\epsilon \to 0} \sup_{\|\delta x\| < \epsilon} \left(\frac{\|\delta f\|}{\|f\|} \Big/ \frac{\|\delta x\|}{\|x\|} \right) \tag{1.9}$$

(relativní číslo podmíněnosti), kde $\delta f \equiv f(x + \delta x) - f(x)$. Řekneme, že úloha je *špatně podmíněná* pokud $\kappa \gg 1$. V opačném případě mluvíme o *dobře podmíněné* úloze.

V případě, že funkční závislost $f(x)$ je diferencovatelná, je číslo podmíněnosti $\hat{\kappa}$ dáno prostě maximem (supremem) velikosti derivace (případně normou jakobiánu zobrazení f).

Typické příklady špatně podmíněných problémů:

- *Vyčíslování funkce v okolí singularity.* Například funkce $f(x) = (\ln 1/x)^{-1/8}$ má limitu 0, pro $x \to 0+$, ale pro $x = 10^{-N}$ je $f(x) = N^{-1/8}$, tj. například pro $N = 100$ je vzdálenost δx od $x = 0$ skutečně hodně malá, ale přesto $f(x) \simeq 0.5073$.

- *Nalezení kořenů polynomu zadaného jeho koeficienty.* Uvažte polynom

  $$p(z) = z^{10} - 10z^9 + 45z^8 - ... - 10z + 1 = (z - 1)^{10},$$

  jenž má jediný desetinásobný kořen $z = 1$. Po záměně koeficientu $a_0 = 1 \to 1 - 10^{-10}$ dostaneme kořeny $z_k = 1 + 0.1 \times e^{2\pi i k/10}$, pro $k = 0, 1, ..., 9$, tj. chyba v určení koeficientu polynomu $p(z)$ se zvětšila o devět řádů (číslo podmíněnosti úlohy v tomto případě dokonce diverguje).

- *Řešení soustav lineárních rovnic.* Později si zvlášť definujeme číslo podmíněnosti matice a uvidíme, že řešení soustavy rovnic pro zvolenou matici soustavy může být extrémně špatně podmíněným problémem.

- *Počáteční úlohy pro diferenciální rovnice* mohou být špatně podmíněné. Uvažme úlohu $y''(x) = y$ s počáteční podmínkou $y'(0) = a$, $y(0) = -a$. Řešením této úlohy je ae^{-x}. Při perturbaci počáteční podmínky musíme vzít v úvahu také druhé lineárně nezávislé řešení e^{+x}, takže číslo podmíněnosti úlohy nalezení řešení v bodě x je řádu e^{2x}.

Na druhé straně mezi dobře podmíněné problémy patří například výpočet určitého integrálu funkce nebo určení vlastních vektorů a vlastních čísel hermitovské matice.

Kapitola 2

Iterace, řešení nelineárních rovnic (a soustav)

Ve fyzice se často setkáváme s potřebou řešit nelineární rovnici typu

$$x = f(x). \tag{2.1}$$

Řešení této rovnice lze hledat pomocí iterací, tj. zvolíme si vhodný počáteční odhad $x = x_0$ a ten zpřesňujeme postupným aplikováním funkce f:

$$x_{n+1} = f(x_n). \tag{2.2}$$

Konvergence této posloupnosti je za určitých okolností zaručena následující větou, kterou byste měli znát z matematické analýzy.

Věta: *O pevném bodě kontrahujícího zobrazení.* Nechť f je spojité zobrazení definované na množině D, kterou zobrazuje do sebe, a nechť f je kontrahující Lipšicovské, tj. existuje $L < 1$ takové, že pro každé $x_1, x_2 \in D$ platí $|f(x_1) - f(x_2)| < L|x_1 - x_2|$. Potom

1. f má na D právě jeden pevný bod: $x_p = f(x_p)$,
2. posloupnost $x_n = f(x_{n-1})$ konverguje k x_p pro každou volbu x_0,
3. $|x_n - x_p| \leq \frac{L^n}{1-L}|x_1 - x_0|$.

Příkladů nalezneme ve fyzice mnoho. Patří mezi ně například *Keplerova rovnice*

$$E = M + \epsilon \sin E,$$

v níž E je neznámá veličina (tzv. excentrická anomálie související s polohou planety na orbitě), $0 < \epsilon < 1$ je známá excentricita orbity a M je zadané číslo související s časem. Pokud vás to zajímá, detailní význam veličin naleznete např. na

`http://en.wikipedia.org/wiki/Kepler_equation#Position_as_a_function_of_time`

Jiným příkladem, v němž $x = \psi$ je vektor z Hilbertova prostoru a f je zobrazení tohoto prostoru do sebe, je Lippmannova-Schwingerova rovnice

$$|\psi\rangle = |\phi\rangle + \hat{G}_0\hat{V}|\psi\rangle.$$

Tuto rovnici lze rovněž za určitých okolností řešit iteracemi, přičemž dostaneme tzv. Bornovu řadu. Jako iterační metodu lze také chápat tzv. metodu selfkonzistentního pole ve fyzice mnoha částic. Zde místo řešení pohybu všech interagujících částic najednou řešíme pohyb každé částice zvlášť v jakémsi pomocném středním poli, ale toto střední pole můžeme určit jedině na základě znalosti trajektorií (resp. vlnových funkcí) jednotlivých částic. Toto střední pole můžeme hledat pomocí iterací.

2.1 Rychlost konvergence a její zvyšování

Definujme chybu n-tého iteračního kroku $e_n \equiv x_n - x_p$. Potom platí

$$e_{n+1} = x_{n+1} - x_p = f(x_n) - f(x_p) = f(x_p + e_n) - f(x_p) \simeq f'(x_p)e_n + \frac{1}{2}f''(x_p)e_n^2 + ..., \quad (2.3)$$

kde jsme předpokládali, že chyba e_n je již malá a použili jsme Taylorova rozvoje funkce $f(x)$ kolem pevného bodu x_p. Nyní rozlišme několik možností

1. *Lineární konvergence:* $f'(x_p) \neq 0$. Potom přibližně platí

   $$e_{n+1} = qe_n$$

   a tedy

   $$e_n \sim q^n,$$

   kde $q = f'(x_p)$. Při konvergenci tohoto typu roste počet správných cifer, na něž známe pevný bod x_p, lineárně s počtem kroků.

2. *Kvadratická konvergence:* $f'(x_p) = 0$. Potom platí

   $$e_{n+1} = \frac{1}{2}f''(x_p)e_n^2$$

   a počet správných cifer aproximace pevného bodu $x_n \simeq x_p$ se v každém kroku zhruba zdvojnásobí.

3. *Pozn.* Pokud $f''(x_p) = f'(x_p) = 0$, jde o konvergenci ještě vyššího řádu (řád bude určen nejnižší nenulovou derivací). Obecně řádem metody nazýváme nenulové číslo α, pro nějž existuje konečná, nenulová limita

   $$\lim_{n\to\infty} \frac{|x_{n+1} - x_p|}{|x_n - x_p|^\alpha}. \quad (2.4)$$

Příklad: Mačkáním klávesy cos na kalkulačce se prakticky přesvědčíte, že řešení algebraické rovnice

$$\cos(x) = x$$

lze nalézt iteracemi a že konvergence je lineární pro libovolnou počáteční hodnotu x_0.

Pro řešení jednoduchých algebraických úloh jako v předcházejícím příkladě je lineární konvergence většinou postačující. Problém nastává, pokud je vyčíslení funkce $f(x)$ časově náročnou úlohou (například nalezení vlastního čísla okrajové úlohy pro parciální diferenciální rovnici). Potom se vyplatí přemýšlet o případném urychlení konvergence. V následujícím si ukážeme

jak toho dosáhnout. Přitom se ukazuje, že můžeme dostat konvergující iterace i pro funkci, která nebyla v okolí pevného bodu kontrahující. Předpokládejme, že posloupnost $x_0, x_1, x_2, ...$ konverguje lineárně k pevnému bodu x_p. Potom můžeme psát

$$x_n - x_p = cq^n, \tag{2.5}$$
$$x_{n+1} - x_p = cq^{n+1}, \tag{2.6}$$
$$x_{n+2} - x_p = cq^{n+2}. \tag{2.7}$$

Tyto rovnice můžeme chápat jako soustavu tří rovnic pro tři neznámé c, q a x_p, za předpokladu, že známe hodnoty čísel posloupnosti x_0, x_1, x_2... Řešení lze najít například následujícím způsobem. Především je ihned vidět:

$$q = \frac{x_{n+1} - x_p}{x_n - x_p} = \frac{x_{n+2} - x_p}{x_{n+1} - x_p},$$

což je už jen jedna rovnice pro x_p. Jejím řešením dostáváme

$$x_p = \frac{x_n x_{n+2} - x_{n+1}^2}{x_{n+2} - 2x_{n+1} + x_n} = x_n - \frac{(x_{n+1} - x_n)^2}{x_{n+2} - 2x_{n+1} + x_n}. \tag{2.8}$$

Rozmyslete si, že zatímco oba tyto výrazy jsou matematicky ekvivalentní, druhé vyjádření vede na menší zaokroulovací chybu (viz předchozí kapitola a poznámky o odčítání skoro stejně velkých čísel). Poslední z výrazů lze jednoduše vyjádřit jako

$$x_p \simeq x'_n \equiv x_n - \frac{(\Delta x_n)^2}{\Delta^2 x_n}, \tag{2.9}$$

kde jsme identifikovali operátor dopředné diference $\Delta x_n \equiv x_{n+1} - x_n$, $\Delta^2 x_n \equiv \Delta(\Delta x_n) = x_{n+2} - 2x_{n+1} + x_n$ (viz následující kapitolu). Navíc jsme si uvědomili, že vzorec je samozřejmě jen přibližný a výsledek závisí na n. Definuje tedy novou posloupnost x'_n. Dá se dokázat, že

$$\lim_{n\to\infty} \frac{x'_n - x_p}{x_n - x_p} = 0$$

a posloupnost x'_0, x'_1, x'_2,... tedy konverguje k x_p rychleji, než původní posloupnost. Obecně platí, že konvergence posloupnosti x'_n je lineární, ale s menším kvocientem q'. Použití vzorce (2.9) se dá tedy opakovat, přičemž získáme novou posloupnost x''_n popřípadě dále $x_n^{(3)}$, $x_n^{(4)}$, ... O takovém urychlování konvergence mluvíme jako o Aitkenově Δ^2 algoritmu.

Ponaučení: Uvedený postup se dá obecně chápat jako extrapolace posloupnosti x_0, x_1, x_2, ... pro $n \to \infty$, přičemž používáme známé asymptotické chování chyby (2.5) pro $n \to \infty$. S tímto postupem se můžeme setkat v numerické matematice častěji a vede většinou k nejefektivnějším algoritmům pro získávání velmi přesných výsledků (viz *Rombergova integrace*, nebo *Bulirsch*ova-*Stoer*ova metoda integrace diferenciálních rovnic; tam se ovšem provádí extrapolace integračního kroku $h \sim 1/n \to 0$).

Další vylepšení (*Aitken-Steffensen*): Výše uvedeným postupem nejdříve vygenerujeme posloupnost x_n iterováním funkce $f(x)$ a potom použitím vzorce (2.9) vytvoříme novou posloupnost x'_n, a dále můžeme pokračovat opakováním vzorce (2.9) a konstruovat x''_n, $x_n^{(3)}$, ... Existuje také alternativní postup:

1. Opět definujeme $x_0^{(0)} \equiv x_0$, $x_1^{(0)} \equiv x_1 = f(x_0)$, $x_2^{(0)} \equiv x_2 = f(x_1)$.

2. Jako v Aitkenově algoritmu vypočteme pro $k = 0, 1, 2, ...$

$$x_0^{(k+1)} = x_0^{(k)} - \frac{[x_1^{(k)} - x_0^{(k)}]^2}{x_2^{(k)} - 2x_1^{(k)} + x_0^{(k)}},$$

3. ale při tom $x_1^{(k+1)} = f(x_0^{(k+1)})$, $x_2^{(k+1)} = f(x_1^{(k+1)})$.

Povšimněte si dobře rozdílu mezi Aitkenovým a Aitkenovým-Steffensenovým algoritmem, který spočívá ve zdánlivě nepodstatném detailu. Zatímco v Aitkenově algoritmu spočítáme celou poloupnost $x_i^{(0)}$ iterováním funkce $f(x)$ a další $x_i^{(k)}$ již generujeme jen vzorcem (2.9), v Aitkenově-Steffensonově algoritmu používáme stále kombinaci iterování funkce $f(x)$ a extrapolace vzorcem (2.9). Ukazuje se, že posloupnost $x_0^{(k)}$, $k = 0, 1, 2, ...$ (zdůrazněme, že se nyní mění horní index místo dolního) díky tomu konverguje kvadraticky k x_p. To je důsledkem následujících tvrzení:

- Platí $x_0^{(k+1)} = \tilde{f}(x_0^{(k)})$, kde

$$\tilde{f}(x) = x - \frac{[f(x) - x]^2}{f(f(x)) - 2f(x) + x}.$$

- Takto definovaná funkce má pevný bod ve stejném x_p jako funkce $f(x)$ a
- pro její derivaci v tomto bodě platí $\tilde{f}'(x_p) = 0$.

První tvrzení je jen jiný přepis Aitkenova-Steffensenova algoritmu a další dvě tvrzení získáme rozvojem $\tilde{f}(x)$ v okolí bodu $x = x_p$. Dosadíme-li rozvoj $f(x_p+\epsilon) = x_p+q\epsilon+O(\epsilon^2)$ a $f(f(x_p+\epsilon)) = x_p + q^2\epsilon + O(\epsilon^2)$ do definice $\tilde{f}(x)$ dostáváme totiž

$$\tilde{f}(x_p + \epsilon) = x_p + \epsilon - \frac{\epsilon^2(q-1)^2}{\epsilon(q^2 - 2q + 1)} + O(\epsilon^2) = x_p + O(\epsilon^2).$$

2.2 Řešení nelineárních algebraických rovnic

Problém: najít řešení $x = x_r$ rovnice

$$g(x) = 0, \tag{2.10}$$

kde $g(x)$ je zadaná reálná funkce reálné proměnné x. Věta o středním bodě zaručuje, že každá spojitá funkce $g(x)$, která na intervalu $\langle a, b\rangle$ mění znaménko, tj. $g(a)g(b) < 0$ nabývá uvnitř tohoto intervalu nulové hodnoty. Numerickou metodu, která za podmínek této věty vždy vede k nalezení kořene s předem zadanou přesností ϵ nazýváme *vždy konvergentní*. Mezi takové metody patří metoda bisekce.

Detailní implementaci **metody bisekce** najdete například v numerických receptech [1]. Vychází z hodnot funkce v krajních bodech intervalu $\langle a, b\rangle$. Tento interval postupně zmenšuje na poloviční tím, že vyčíslí funkční hodnotu v prostředním bodě intervalu a vybere tu část na níž zůstane splněna podmínka $g(a)g(b) < 0$. Délka intervalu, udávající chybu odhadu polohy kořene, se postupně mění v každém kroku na polovinu a rychlost konvergence metody je tudíž lineární.

Iterační metody: Nechť $\phi(x)$ je funkce taková, že $0 < |\phi(x)| < \infty$ pro všechna $x \in \langle a, b\rangle$. Potom rovnice

$$x = f(x) \equiv x - \phi(x)g(x)$$

má stejné kořeny jako $g(x) = 0$. Jednotlivé iterační metody se liší volbou $\phi(x)$:

- $\phi(x) = m$ =konst. Potom můžeme hledat kořen x_r iterováním funkce $f(x)$ podle věty o pevném bodě kontrahujícího zobrazení, platí-li $|f'(x)| < 1$, tj. $0 < mg'(x) < 2$ pro všechna $x \in \langle a, b \rangle$.

- *Newtonova metoda.* Podmínka pro kvadratickou konvergenci zní

$$f'(x_r) = 1 - \phi'(x_r)g(x_r) - \phi(x_r)g'(x_r) = 1 - \phi(x_r)g'(x_r) = 0,$$

  kde jsme využili toho, že $g(x_r) = 0$. Kvadraticky konvergentní metodu tedy dostaneme položením $\phi(x) = 1/g'(x)$. To vede na konstrukci postupných aproximací řešení x_r

$$x_{n+1} = x_n - \frac{g(x_n)}{g'(x_n)}.$$

  Použitelnost metody: Konvergence $x_n \to x_r$ samozřejmě není zaručena pro libovolnou volbu x_0. Ale z předchozího je zřejmé, že pokud je $|f'(x)| < 1$ na nějakém intervalu, je $f(x)$ na tomto intervalu kontrahujícím zobrazením a můžeme volit libovolné x_0 z tohoto intervalu. Nalézt takový interval může být pro některé funkce $g(x)$ přinejmenším obtížné. Pokud však má funkce $g(x)$ spojitou nenulovou derivaci v bodě x_r máme alespoň zaručeno splnění této podmínky v nějakém okolí bodu x_r. Prakticky lze kombinovat metodu bisekce pro hrubou lokalizaci kořene s Newtonovou metodou pro rychlé nalezení přesné hodnoty. (Poznámka: pokud jste se s Newtonovou metodou ještě nesetkali, rozmyslete si její geometrický význam, viz též [1]).

- *Modifikovaná Newtonova metoda - metoda sečen.* V praxi se často setkáme s případem, kdy derivaci funkce $g(x)$ neznáme. Potom můžeme spočítat derivaci přibližně jako

$$g'(x_n) \simeq [g(x_n) - g(x_{n-1})]/(x_n - x_{n-1})$$

  což vede na posloupnost iterací

$$x_{n+1} = \frac{x_{n-1}g(x_n) - x_n g(x_{n-1})}{g(x_n) - g(x_{n-1})}.$$

  Dá se ukázat, že pokud jsou g' a g'' spojité, potom existuje okolí bodu x_r takové, že posloupnost x_n konverguje k x_r pro libovolné x_0 z tohoto okolí. Přitom konvergence je řádu $\alpha = (\sqrt{5} + 1)/2 \simeq 1.618$ [1, 5, 9].

Příklad: Jednoduchým příkladem použití Newtonovy metody je klasický algoritmus pro výpočet druhé odmocniny. Volíme-li $g(x) = x^2 - c$, dostaneme iterace

$$x_{n+1} = f(x_n) = \frac{1}{2}\left(x_n + \frac{c}{x_n}\right),$$

přičemž se dá dokázat, že posloupnost vždy konverguje. Například pro výpočet $\sqrt{2}$ s počátečním odhadem $x_0 = 1$ dostáváme postupně aproximace 1.5, 1.4166, 1.4142156, 1.4142135623746, 1.41421356237309504880168896, t.j. po 5 iteracích dostáváme výsledek s chybou nepřevyšující strojové ϵ pro dvojitou přesnost.

Poznámka: Ačkoli máme zajištěno, že existuje nějaké okolí hledaného kořene x_r, v němž Newtonova metoda konverguje kvadraticky, může být obtížné zjistit, v jaké oblasti přesně metoda konverguje. Situace je ještě komplikovanější, pokud existuje několik kořenů. Například pro

funkci $g(x) = x^3 - 1$ existují tři komplexní kořeny. Nyní je možno každý bod komplexní roviny $x_0 \in \mathbb{C}$ obarvit jednou ze tří barev podle toho, ke kterému kořenu konverguje Newtonova metoda startující z bodu x_0. Výsledkem je překvapivě komplikovaný obrazec nazývaný *Newtonův fraktál*, který si můžete prohlédnout například na

`http://en.wikipedia.org/wiki/Newton_fractal`

2.3 Iterace a algebraické rovnice ve více proměnných

Zobecnění Banachovy věty o pevném bodě kontrahujícího zobrazení do více dimenzí je přímočaré. Stačí považovat x za vektor a $f(x)$ za vektorovou funkci vektorové proměnné, tj.

$$x_i^{n+1} = f_i(x_1^n, x_2^n, ..., x_D^n), \qquad \text{pro } i = 1, 2, ..., D, \tag{2.11}$$

kde D je dimenze prostoru, a dále absolutní hodnotu $|x|$ ve větě nahradíme normou vektoru $\|x\|$. Také jsme trochu modifikovali značení. Horní index ve výrazu x^n nebo x_i^n čísluje jednotlivé členy posloupnosti vektorů, zatímco dolní index jsme nyní vyhradili indexu číslujícímu složky vektoru.

Analýzu chyb je oproti jednodimenzionálnímu případu nutno mírně modifikovat. Nyní můžeme psát

$$e^{n+1} = x^{n+1} - x^p = f(x^n) - f(x^p) = J(x^p)e^n + O(\|e^n\|^2), \tag{2.12}$$

kde $J(x)$ je matice jakobiánu zobrazení $f(x)$, tj.

$$J = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_D} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_D} \\ \vdots & \vdots & & \vdots \\ \frac{\partial f_D}{\partial x_1} & \frac{\partial f_D}{\partial x_2} & \cdots & \frac{\partial f_D}{\partial x_D} \end{pmatrix}.$$

Má-li $f(x)$ být kontrahující zobrazení, musí být všechna vlastní čísla matice $J(x)$ menší než 1 pro všechna x z uvažované oblasti. Pokud je matice $J(x_p)$ nenulová, dostáváme lineární konvergenci, neboli konvergenci prvního řádu. Obecně lze řád konvergence definovat opět vzorcem (2.4) v němž nahradíme absolutní hodnotu normou vektoru. Speciálně nás bude zajímat možnost konstruovat metody vyšších řádů. Aitkenův Δ^2-algoritmus bohužel nelze přímočaře zobecnit, protože z několika následujících členů posloupnosti není snadné eliminovat celou neznámou matici jakobiánu. Pokud však umíme matici jakobiánu zkonstruovat jiným způsobem, podaří se nám zobecnit alespoň Newtonovu metodu.

Při *řešení soustav algebraických* rovnic budeme sledovat postup z přechozí kapitoly. Naším cílem je opět nalézt takové x^r (tentokrát vektor), který splňuje soustavu algebraických rovnic

$$g(x^r) = 0.$$

Uvědomme si, že $g(x)$ je nyní vektorová funkce vektorové proměnné reprezentovaná komponentami $g_i(x_1, x_2, ..., x_D)$, $i = 1, 2, ..., D$. Nechť pro všechna x z nějaké množiny M je $A(x)$ nesingulární matice. Potom $x^r \in M$ splňuje naši soustavu právě tehdy, když x^r je pevným bodem zobrazení

$$f(x) = x - A(x)g(x).$$

Nyní přicházejí v úvahu různé volby matice A. Podobně jako v jednodimenzionálním případě je možno vzít vhodnou konstantní matici, ale může být obtížné ji zvolit tak, aby $f(x)$ bylo kontrahující zobrazení. Na druhé straně požadavek, aby Jacobiho matice funkce $f(x)$ byla nulová v bodě x^r vede ke vztahu

$$0 = \frac{\partial f_i}{\partial x_j} = \delta_{ij} - \sum_k \left(\frac{\partial A_{ik}}{\partial x_j} g_k + A_{ik} \frac{\partial g_k}{\partial x_j} \right).$$

První člen v sumě přes k je nulový v bodě $x = x^r$ a dostaneme tedy podmínku, že matice $A_{ik}(x)$ musí být inverzní k matici jakobiánu J_g zobrazení $g(x)$ v bodě $x = x^r$ a můžeme tedy položit $A(x) = J_g(x)^{-1}$. Nyní máme pohromadě všechny komponenty ke konstrukci iteračního schematu *Newtonovy metody* ve více dimenzích

$$x^{n+1} = x^n - J_g(x^n)^{-1} g(x^n). \tag{2.13}$$

Příklad: (Transformace od kartézských k polárním souřadnicím). Způsob fungování Newtonovy metody ve více dimenzích si ukážeme tak, že vyřešíme soustavu rovnic

$$\begin{aligned} r\cos\phi - x_0 &= 0, \\ r\sin\phi - y_0 &= 0, \end{aligned}$$

kde x_0, y_0 jsou zadaná reálná čísla a r, ϕ hledané neznámé. Tuto soustavu lze opět chápat jako rovnici ve tvaru $g(x) = 0$, kde vektor $x = (r, \phi)$ a funkce $g(x)$ má komponenty $g_1(r, \phi) = r\cos\phi - x_0$ a $g_2(r, \phi) = r\sin\phi - y_0$. Matici jakobiánu a její inverzi nyní snadno najdeme

$$J(r,\phi) = \begin{pmatrix} \cos\phi & -r\sin\phi \\ \sin\phi & r\cos\phi \end{pmatrix}, \qquad J(r,\phi)^{-1} = \frac{1}{r} \begin{pmatrix} r\cos\phi & r\sin\phi \\ -\sin\phi & \cos\phi \end{pmatrix}$$

a iterační vzorec tedy je

$$\begin{aligned} r^{n+1} &= x_0 \cos\phi^n + y_0 \sin\phi^n, \\ \phi^{n+1} &= \phi^n - (x_0 \sin\phi^n - y_0 \sin\phi^n)/r^n. \end{aligned}$$

Můžete si sami ověřit na počítači, že podobně jako v algoritmu pro výpočet druhé odmocniny dostáváme pro rozumnou volbu počáteční podmínky správný výsledek na plný počet míst během řádově pěti iterací.

Nakonec ještě několik praktických poznámek k implementaci iteračních algoritmů. Při praktické použití vzorce (2.11) je potřeba pamatovat na uložení celého vektoru x^n do pomocné proměnné, než jeho složky začneme přepisovat novými hodnotami x_i^n. Tomu se vyhneme, pokud schéma

$$\begin{aligned} x_1^{n+1} &= f_1(x_1^n, x_2^n, ..., x_D^n), \\ x_2^{n+1} &= f_2(x_1^n, x_2^n, ..., x_D^n), \\ x_3^{n+1} &= f_3(x_1^n, x_2^n, ..., x_D^n), \\ &\vdots \\ x_D^{n+1} &= f_D(x_1^n, x_2^n, ..., x_D^n) \end{aligned} \qquad \text{nahradíme} \qquad \begin{aligned} x_1^{n+1} &= f_1(x_1^n, x_2^n, ..., x_D^n), \\ x_2^{n+1} &= f_2(x_1^{n+1}, x_2^n, ..., x_D^n), \\ x_3^{n+1} &= f_2(x_1^{n+1}, x_2^{n+1}, ..., x_D^n), \\ &\vdots \\ x_D^{n+1} &= f_D(x_1^{n+1}, x_2^{n+1}, ..., x_{D-1}^{n+1}, x_D^n). \end{aligned}$$

Tím jsme samozřejmě změnili algoritmus a výsledné iterace už neodpovídají vzorci (2.11). Nicméně nový vzorec má stejný pevný bod a jak si ukážeme později v souvislosti s řešením

okrajových úloh pro diferenciální rovnice, konverguje výsledný algoritmus často rychleji. V této souvislosti mluvíme o relaxaci. Další možností je modifikovat vzorec (2.11) následovně

$$x_i^{n+1} = (1-\omega)x_i^n + \omega f_i(x_1^n, x_2^n, ..., x_D^n), \qquad \text{pro } i = 1, 2, ..., D, \tag{2.14}$$

kde ω je takzvaný relaxační parametr. Takto modifikované iterace mají opět stejný bod jako ty původní a dodatečnou volnost ve volbě parametru ω lze použít k optimalizaci konvergence, jak si povíme více u popisu iteračních metod řešení soustav lineárních rovnic. Také tento nový vzorec lze samozřejmě použít v relaxované podobě

$$x_i^{n+1} = (1-\omega)x_i^n + \omega f_i(x_1^{n+1}, ..., x_{i-1}^{n+1}, x_i^n, ..., x_D^n), \qquad \text{pro } i = 1, 2, ..., D. \tag{2.15}$$

Kapitola 3

Interpolace a aproximace funkcí

Mezi základní úlohy numerické matematiky patří diskrétní reprezentace spojitých funkcí. Jednou z možností je nahrazení funkce, kterou chceme reprezentovat, funkcí z nějaké množiny, kterou lze charakterizovat konečným počtem parametrů (reálných čísel). Speciálním případem je lineární vektorový prostor získaný jako lineární obal konečné množiny funkcí. K nejvýznamnějšímu případu tohoto typu aproximace patří aproximace pomocí polynomů do řádu n. Při aproximaci funkce $f(x)$, kterou nahradíme její reprezentací $\bar{f}(x)$, je potřeba nějak charakterizovat chybu aproximace $\Delta f(x) \equiv f(x) - \bar{f}(x)$. V následujících odstavcích se budeme bavit o aproximacích ve smyslu nejmenších čtverců (minimalizace $\int |\Delta f(x)|^2$) a Čebyševově aproximaci (minimalizace $\max |\Delta f(x)|$). Začneme však nesmírně užitečným pojmem interpolace funkce.

3.1 Polynomiální interpolace

O polynomiální interpolaci mluvíme v případě, že nahrazujeme funkci $f(x)$ procházející body (x_i, f_i), $i = 0, 1, ..., n$ polynomem

$$L_n(x) \equiv a_0 + a_1 x + ... + a_n x^n,$$

který rovněž prochází těmito body, tj. $L_n(x_i) = f_i$. Tuto podmínku lze napsat v maticovém tvaru:

$$\mathbf{Va} \equiv \begin{pmatrix} 1 & x_0 & x_0^2 & ... & x_0^n \\ 1 & x_1 & x_1^2 & ... & x_1^n \\ \vdots & \vdots & \ddots & & \vdots \\ 1 & x_n & x_n^2 & ... & x_n^n \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} f_0 \\ f_1 \\ \vdots \\ f_n \end{pmatrix}, \tag{3.1}$$

který představuje soustavu $(n + 1)$ lineárních rovnic pro neznámé koeficienty a_i sdružené do vektoru $\mathbf{a}$. Matice soustavy, tvořená „mocninami“ sloupcového vektoru

$$\mathbf{x} \equiv (x_0, x_1, ..., x_n)^T,$$

se nazývá *Vandermode*ho matice a snadno nahlédneme, že její determinant je

$$|\mathbf{V}| = \prod_{i>j} (x_i - x_j).$$

Na determinant matice V můžeme totiž pohlížet jako na polynom v proměnných x_0, x_1, ..., x_n, který má nuly v bodech, kde vektory tvořící jeho řádky jsou lineárně závislé, tj. právě

když $x_i = x_j$ pro nějaké i, j. Výše uvedená formule pak je rozvojem tohoto polynomu do jeho $n(n+1)/2$ kořenových činitelů. Současně je zřejmé, že pokud jsou všechny body x_i navzájem různé, potom je determinant matice $\mathbf{V}$ různý od nuly a soustava (3.1) má právě jedno řešení.

Dokázali jsme existenci a jednoznačnost interpolačního polynomu $L_n(x)$. O tomto polynomu mluvíme jako o *Lagrange*ově interpolačním polynomu. Tento polynom můžeme nalézt řešením výše uvedené soustavy. Zmiňme ještě **dvě** jiné **metody** konstrukce tohoto polynomu. **První** z nich se většinou používá při teoretické analýze a důkazech a spočívá v rozvoji do baze $l_i(x)$ v prostoru polynomů do řádu n, která má příhodnou vlastnost

$$l_i(x_j) = \delta_{ij}.$$

Rozvoj Lagrangeova interpolačního polynomu do této baze zjevně je

$$L_n(x) = \sum_{i=0}^{n} f_i l_i(x). \tag{3.2}$$

Úlohu jsme tedy redukovali na nalezení $l_i(x)$. Snadno se přesvědčíte, že polynomy s touto vlastností lze psát jako

$$\begin{aligned} l_i(x) &= \left.\frac{(x-x_0)(x-x_1)...(x-x_n)}{(x_i-x_0)(x_i-x_1)...(x_i-x_n)}\right| \begin{array}{l}\text{s vynecháním i-tého členu}\\ \text{v čitateli i jmenovateli}\end{array} \\ &\equiv \frac{\omega_{n+1}(x)}{(x-x_i)\omega'_{n+1}(x_i)}, \end{aligned}$$

kde

$$\omega_{n+1}(x) = (x-x_0)(x-x_1)...(x-x_n).$$

Druhá metoda konstrukce interpolačního polynomu je vhodná pro praktický výpočet interpolačního polynomu v nějakém bodě x a spočívá v postupném výpočtu sloupců následující tabulky (*Aitkenovo-Nevillovo schema*)

$$\begin{array}{llllll} x_0 & f_0 = P_0^{(0)} & & & & \\ x_1 & f_1 = P_0^{(1)} & P_1^{(01)} & & & \\ x_2 & f_2 = P_0^{(2)} & P_1^{(12)} & P_2^{(012)} & & \\ x_3 & f_3 = P_0^{(3)} & P_1^{(23)} & P_2^{(123)} & P_3^{(0123)} & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{array}$$

V jednotlivých sloupcích konstruujeme postupně polynomiální aproximace vyšších řádů, přičemž například $P_2^{(012)}$ je hodnota (v bodě x) polynomu 2. řádu procházejícího body (x_0, f_0), (x_1, f_1), (x_2, f_2). Při konstrukci následujícího sloupce tabulky využíváme postupně fromule

$$P_{j+1}^{(i,i+1,...,i+m)} = \frac{(x-x_{i+m})P_j^{(i,i+1,...,i+m-1)} + (x_i-x)P_j^{(i+1,...,i+m-1,i+m)}}{x_i - x_{i+m}}.$$

Platnost této formule snadno nahlédneme, když si uvědomíme, že jde o vážený průměr dvou sousedních polynomů a výsledný polynom proto musí procházet všemi vnitřními body $i+1, ..., i+m-1$, kterými prochází oba polynomy. Konkrétní volba váhových faktorů $\frac{x-x_{i+m}}{x_i-x_{i+m}}$ a $\frac{x_i-x}{x_i-x_{i+m}}$ zajistí, že výsledný polynom prochází i krajními body i a $i+m$.

Z matematické analýzy známe Weierstrassovu aproximační větu, která říká, že libovolnou spojinou funkci lze libovolně přesně aproximovat nějakým polynomem dost vysokého řádu. Ten ovšem nemusí být shodný s interpolačním polynomem.

Přesnost polynomiální interpolace řeší následující věta (Lagrangeův zbytek): Nechť x_0, x_1, ..., x_n jsou navzájem různé body z intervalu $\langle a, b\rangle$ a $f \in C^{n+1}\langle a, b\rangle$. Pak pro každé $x \in \langle a, b\rangle$ existuje $\xi \in \langle a, b\rangle$, tak že

$$f(x) - L_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!}\omega_{n+1}(x).$$

Podobnost se zbytkem v Taylorově rozvoji je nápadná a nepřekvapí, že například pokud má funkce $f(x)$ Taylorův rozvoj v některém z krajních bodů intervalu $\langle a, b\rangle$ a tento rozvoj konverguje na celém tomto intervalu, pak lze výše uvedený zbytek odhadnout zhora číslem $\max|f^{(n+1)}||b-a|^{n+1}/(n+1)!$, které konverguje k nule, a tudíž $L_n(x)$ konverguje k $f(x)$. Obecně je potřeba jisté ostražitosti neboť chování $L_n(x)$ pro velká n silně závisí na volbě množiny x_i. Klasickým příkladem je aproximace funkce

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

která má na celé reálné ose spojité všechny derivace, ale například pro rovnoměrně rozloženou síť bodů na intervalu $\langle -5, 5\rangle$ posloupnost $L_n(x)$ diverguje (viz obrázek - dodám později). To lze snadno pochopit, když si člověk uvědomí, že funkce má póly v bodech $x = \pm i$ a Taylorova řada v bodě 0 má poloměr konvergence 1, což nepokrývá celý interval $\langle -5, 5\rangle$. Naproti tomu se dá ukázat, že pokud volíme x_i na intervalu $\langle -1, 1\rangle$ rozloženy podle vzorce

$$x_i = \cos\left(\frac{\pi}{2} \cdot \frac{2i+1}{n+1}\right), \qquad i = 0, 1, ..., n$$

potom $|\omega_n(x)| < 2^{-(n-1)}$ a navíc Lagrangeův interpolační polynom libovolné funkce $f(x)$ spojité na tomto intervalu konverguje stejnoměrně na tomto intervalu k $f(x)$ pro $n \to \infty$. K tomuto tématu se ještě vrátíme níže v povídání o Čebyševových polynomech.

3.2 Hermiteova interpolace

Zobecněním Lagrengeovy interpolace je interpolace Hermiteova. Zde požadujeme, aby se aproximovaná funkce $f(x)$ a interpolační polynom $H_m(x)$ shodovali v navzájem různých bodech x_i, a to včetně derivací do řádu $\alpha_i - 1$. Polynom $H_m(x)$ je opět určen jednoznačně, pokud se počet kladených podmínek $\alpha_0 + \alpha_1 + ... + \alpha_n$ a počet koeficientů polynomu $(m+1)$ shodují.

Přesnost Hermiteovy interpolace: Za stejných pomínek jako pro Lagrangeovu interpolaci a pro $f \in C^{m+1}\langle a, b\rangle$ máme

$$f(x) - H_m(x) = \frac{f^{(m+1)}(\xi)}{(m+1)!}\omega_{m+1}(x),$$

kde

$$\omega_{m+1}(x) = (x - x_0)^{\alpha_0}(x - x_1)^{\alpha_1}...(x - x_n)^{\alpha_n}.$$

Podobnost Lagrangeova zbytku s Taylorovou větou se nyní osvětluje, neboť aproximace jak Taylorovým tak Lagrangeovým polynomem je obsažena ve výše uvedeném vzorci jako speciální

případ. Hermiteův interpolační polynom lze opět nalézt řešením jisté soustavy lineárních rovnic, nebo přímým vzorcem, který pro pozdější referenci uvádíme jen pro případ $\alpha_0 = .. = \alpha_n = 2$. V tom případě platí

$$H_{2n+1}(x) = \sum_{i=0}^{n} s_i(x) f(x_i) + \sum_{i=0}^{n} t_i(x) f'(x_i),$$

Kde s_i a t_i s vlastností $s_i(x_j) = \delta_{ij}$, $s_i'(x_j) = 0$, $t_i(x_j) = 0$, $t_i'(x_j) = \delta_{ij}$ tvoří bazi v prostoru polynomů stupně nejvýše $m = 2n + 1$ a dají se explicitně napsat pomocí polynomů $l_i(x)$ z Lagrangeovy interpolace

$$\begin{aligned} s_i(x) &= [1 - 2(x - x_i) l_i'(x_i)]\, l_i^2(x), \\ t_i(x) &= (x - x_i)\, l_i^2(x). \end{aligned}$$

3.3 Interpolace funkcí na rovnoměrné síti

Speciálním, často se vyskytujícím případem je situace, kdy známe funkci tabelovanou na rovnoměrné síti bodů $x_i = i\,h$, kde $h > 0$ je (většinou v nějakém smyslu malé) reálné číslo. Pro takto tabulovanou funkci se dá odvodit spousta užitečných vzorců pro polynomiální interpolaci, její derivaci a integraci. Některé z těchto vzorců budeme v dalším hojně užívat, ukážeme si proto mocný nástroj pro elegantní odvozování vzorců tohoto typu. Postup bude následující: zavedeme určitou množinu operátorů nad prostorem polynomů a ukážeme jejich vztah k derivování a integrování. Výsledné vztahy budou obsahovat jen hodnoty polynomu v diskrétních bodech. Pokud tyto hodnoty nahradíme hodnotami zkoumané funkce v těchto bodech, můžeme výsledek interpretovat jako vztah pro interpolační polynom proložený těmito hodnotami.

V následujícím budeme předpokládat, že $p(x)$ je polynomem nějakého konečného stupně. Pokud budeme chtít speciálně zdůraznit, že stupeň polynomu je nějaké konkrétní číslo n, budeme psát $p_n(x)$. Můžeme definovat následující operátory

$$\begin{aligned} Ip(x) &= p(x) && \text{identita,} \\ Ep(x) &= p(x + h) && \text{posunutí,} \\ \Delta p(x) &= p(x + h) - p(x) && \text{dopředná diference,} \\ \nabla p(x) &= p(x) - p(x - h) && \text{zpětná diference,} \\ \delta p(x) &= p(x + h/2) - p(x - h/2) && \text{centrovaná diference.} \end{aligned}$$

Všechna tato zobrazení jsou lineární operátory nad prostorem polynomů. Nyní budeme hledat vztahy mezi těmito a dalšími operátory, přičemž rovností dvou operátorů budeme rozumět to, že se rovnají výsledky použití těchto operátorů na libovolný polynom konečného stupně. Jako obvykle definujeme mocninu operátoru jako opakované použití operátoru, tj. například $A^2 p(x) \equiv A(A(p(x)))$. Dále dodefinujme $A^0 \equiv I$ a A^{-1} je takový operátor, že $A^{-1}(A(p(x))) = p(x)$, pro každý polynom $p(x)$. Nyní můžeme definovat libovolnou funkci operátoru pomocí Taylorova rozvoje, například

$$\exp(\Delta) = I + \Delta + \frac{1}{2}\Delta^2 + \frac{1}{2.3}\Delta^3 + ...$$

Základní vztahy. Je dobré si uvědomit, že operátory diference mají jednu důležitou vlastnost

$$\Delta^m p_n(x) = \nabla^m p_n(x) = \delta^m p_n(x) = 0, \qquad \text{pokud } m > n$$

a tedy například pro polynomy druhého řádu je $\exp(\Delta) = I + \Delta + \frac{1}{2}\Delta^2$. Tato vlastnost bude za chvíli klíčová pro interpretaci formulí, které odvodíme. Mocniny operátorů diference jsou úzce svázány s Pascalovým trojúhelníkem. Platí totiž

$$Ep(x) = p(x+h) = p(x+h) - p(x) + p(x) = (\Delta + I)p(x)$$

neboli

$$\boxed{E = I + \Delta}.$$

Pro operátor Δ pak podle binomické věty dostáváme

$$\Delta^n = (E - I)^n = \sum_{k=0}^{n} (-1)^k \binom{n}{k} E^{n-k}.$$

Toho lze využít pro výpočet diferencí funkce s hodnotami f_i tabelované v bodech x_i. Platí například

$$\begin{array}{rcl} \Delta f_i & = & f_{i+1} - f_i \\ \Delta^2 f_i & = & f_{i+2} - 2f_{i+1} + f_i \\ \Delta^3 f_i & = & f_{i+3} - 3f_{i+2} + 3f_{i+1} - f_i \\ \Delta^4 f_i & = & f_{i+4} - 4f_{i+3} + 6f_{i+2} - 4f_{i+1} + f_i \\ & \vdots & \end{array}$$

Podobné vzorce lze odvodit pro zpětnou a symetrickou diferenci.

Interpolační formule. Zobecněním předchozího postupu dostáváme *Newtonovu interpolační formuli*

$$\begin{array}{rcl} f(x+sh) & = & E^s f(x) = (I + \Delta)^s f(x) \\ & = & \left[I + \pi_0(s)\Delta + \dfrac{\pi_1(s)}{2!}\Delta^2 + \cdots + \dfrac{\pi_k(s)}{(k+1)!}\Delta^{k+1} \right] f(x), \end{array}$$

kde

$$\pi_n(s) \equiv s(s-1)...(s-n)$$

přičemž jsme použili Taylorova rozvoje funkce $(1+x)^s$. Striktně řečeno tento vzorec platí pouze pro polynomy $f(x) = p_n(x)$, kde se redukuje na konečný počet členů, neboť, jak jsme zmínili výše je $\Delta^{n+1} p_n(x) = 0$. Pokud jej použijeme na $f(x_i + sh)$ a ořežeme pravou stranu na konečný počet členů s posledním členem $k = n - 1$, zbude na pravé straně polynom stupně n v proměnné s. Koeficienty tohoto polynomu jsou jednoznačně určeny hodnotami $\Delta^k f(x_i)$ tj. pouze hodnotami funkce v bodech x_i, x_{i+1}, ..., x_{i+n+1}. Hodnota $f(x + sh)$ je tedy shodná s hodnotou Lagrangeova interpolačního polynomu proloženého těmito body. Stejný postup jakým jsme odvodili Newtonovu interpolační formuli pomocí dopředné diference, můžeme použít i pro zpětnou diferenci. Výsledek se liší jen znaménky u lichých mocnin ∇^k.

Souvislost s derivováním. Kromě výše zavedených operátorů I, E, Δ, ∇, δ zaveďme ještě operátor D, který polynomu $p(x)$ přiřadí jeho derivaci: $Dp(x) = p'(x)$. Použitím Taylorova rozvoje

$$p(x+h) = p(x) + hp'(x) + \frac{h^2}{2!}p''(x) + \frac{h^3}{3!}p'''(x) + ...$$

dostaneme relaci

$$E = 1 + hD + \frac{(hD)^2}{2!} + \frac{(hD)^3}{3!} + ... \equiv e^{hD}$$

a tedy

$$\boxed{e^{hD} = E = I + \Delta} \qquad \text{neboli} \qquad \boxed{hD = \ln E = \ln(I + \Delta)}.$$

Logaritmus operátoru v posledním z uvedených vzorců je definován Taylorovým rozvojem

$$hD = \Delta - \frac{1}{2}\Delta^2 + \frac{1}{3}\Delta^3 - ... + (-1)^{k+1}\frac{1}{k}\Delta^k ... \tag{3.3}$$

Když si uvědomíme, že po aplikaci na polynom končí pravá strana po konečném počtu členů a obsahuje hodnotu polynomu jen v konečném počtu bodů, můžeme tuto formuli použít k derivování funkce zadané hodnotami v bodech v pravidelné síti. Formuli potom interpretujeme jako derivaci interpolačního polynomu proloženého body funkce. Podobně jako v případě Newtonovy interpolační formule, můžeme postup zopakovat s použitím zpětné diference a výsledek se liší jen znaménky u sudých mocnin diferenčního operátoru. Ještě rychlejší konvergence lze dosáhnout použitím symetrické diference. Symetrická diference δ pracuje s body v polovině mezi body sítě. Alternativně můžeme pracovat s

$$\Delta + \nabla = \exp(hD) - \exp(-hD) = 2\sinh(hD)$$

a tedy

$$hD = \sinh^{-1}(\frac{\Delta + \nabla}{2}) = \sum_{k=0}^{\infty} \frac{(-1)^k (2k)!}{4^k (k!)^2 (2k+1)} \left(\frac{\Delta + \nabla}{2}\right)^{2k+1} = \frac{\Delta + \nabla}{2} - \frac{1}{6}\left(\frac{\Delta + \nabla}{2}\right)^3 + ... \tag{3.4}$$

Všimněte si, že vzorec pro derivaci se symetrickou diferencí obsahuje pouze liché mocniny diferenčního operátoru a po ořezání na konečný počet členů je tedy o řád přesnější než vzorec používající Δ nebo ∇.

Souvislost s integrováním. Pro nalezení formulí pro integrování interpolačního polynomu funkce zadané hodnotami na pravidelné sítí definujeme ještě operátor

$$Jp(x) \equiv \int_x^{x+h} p(t)dt.$$

Platí

$$\begin{aligned} Jp(x) &= h\int_0^1 E^s p(x)ds = h\int_0^1 E^s dsp(x) = h\int_0^1 e^{s\ln E} dsp(x) \\ &= h\frac{E-I}{\ln E}p(x) = \frac{h\Delta}{\ln(I+\Delta)}p(x) \end{aligned}$$

a tedy

$$\boxed{J = \Delta/D} \qquad \text{neboli} \qquad \boxed{JD = DJ = \Delta}.$$

Pokud chceme nalézt integrační formule interpolačního polynomu funkce, musíme udělat Taylorův rozvoj výrazu $h\Delta/\ln(1+\Delta)$ v proměnné Δ. Taylorův rozvoj jmenovatele , který jsme již zmínili výše, napíšeme ve tvaru $\ln(1+\Delta) = \Delta(I - R)$, kde

$$R = \frac{1}{2}\Delta - \frac{1}{3}\Delta^2 + \frac{1}{4}\Delta^3 - ...$$

a tedy

$$\begin{aligned} J &= \frac{h\Delta}{\ln(1+\Delta)} = \frac{h}{I-R} = h(I+R+R^2+R^3...), \\ &= h(I+\frac{1}{2}\Delta-\frac{1}{12}\Delta^2+\frac{1}{24}\Delta^3-\frac{19}{720}\Delta^4+...). \end{aligned}$$

Při odvození předchozího vzorce jsme nejprve využili toho, že součet geometrické řady s kvocientem $q = R$ je $(I-R)^{-1}$ a při přechodu k druhému řádku je prostě potřeba roznásobit členy geometrické s R dosazeným z definice. Použitím tohoto vzorce na polynomy prvního řádu zůstanou na pravé straně pouze dva členy. Pokud vzorec aplikujeme na funkci $f(x)$ v bodě $x = x_0$ dostaneme

$$Jf(x)|_{x=x_0} = \int_{x_0}^{x_1} f(x)dx = \frac{h}{2}[f_0+f_1], \tag{3.5}$$

což je známé *lichoběžníkové pravidlo*. Stejně aplikací na polynomy druhého řádu dostáváme

$$Jf(x)|_{x=x_0} = h[\frac{5}{12}f_0+\frac{2}{3}f_1-\frac{1}{12}f_2].$$

Toto je použitelný vzorec, ale uvědomme si co dostáváme. Levá strana je integrálem interpolačního polynomu proloženého funkcí $f(x)$ bodech x_0, x_1, x_2 v mezích od x_0 do x_1. Na pravé straně se vyskytuje též funkční hodnota f_2 v bodě x_2, jenž leží vně oboru integrace. Abychom odstranili tuto asymetrii, přičteme ještě integrál téhož interpolačního polynomu v intervalu od x_1 do x_2. Ten dostaneme stejným vzorcem s prohozením koeficientů u f_0 a f_2 (můžeme si představovat, že použijeme stejnou integrační formuli pro funkci $f(x_2-x)$) a tedy

$$\int_{x_0}^{x_2} f(x)dx = \frac{h}{3}[f_0+4f_1+f_2], \tag{3.6}$$

což je formule známá pod jménem *Simpsonovo pravidlo*. K vzorcům vyššího řádu se ještě vrátíme, ale nejdříve se vrátíme k lichoběžníkovému pravidlu. Pokud použijeme tento odhad integrálu na několik intervalů následujících po sobě dostaneme tzv. *rozšířené lichoběžníkové pravidlo*, které spočívá v nahrazení integrálu funkce integrálem lomené čáry procházející danými body sítě. Následující důležitá věta hovoří o chybě této aproximace.

Eulerova-Maclaurinova sumační formule: Nechť $f(x) \in C^{2k+2}\langle x_0, x_p\rangle$, potom platí

$$\begin{aligned} \sum_{j=0}^{p} f_j - \frac{1}{2}[f_0+f_p] &= \frac{1}{h}\int_{x_0}^{x_0+ph} f(x)dx \\ &+a_1[f'(x_0+hp)-f'(x_0)]h \\ &+a_2[f^{(3)}(x_0+hp)-f^{(3)}(x_0)]h^3+... \\ &+a_k[f^{(2k-1)}(x_0+hp)-f^{(2k-1)}(x_0)]h^{2k-1}+O(h^{2k+1}), \end{aligned} \tag{3.7}$$

kde a_k jsou konstanty (například $a_1 = \frac{1}{12}$, $a_2 = -\frac{1}{720}$, $a_3 = \frac{1}{30240}$).

Náznak důkazu: Vyšetřeme nejdříve integrál vystupující v dokazované formuli

$$\begin{aligned} \int_x^{x+ph} f(t)dt &= \int_x^{x+h} f(t)dt + \int_{x+h}^{x+2h} f(t)dt + ... + \int_{x+(p-1)h}^{x+ph} f(t)dt = \\ &= [E^0+E^1+E^2+...+E^{(p-1)}]Jf(x) = \frac{E^p-E^0}{E-I}Jf(x) = [E^p-I]D^{-1}f(x), \end{aligned}$$

kde jsme využili výrazu pro součet geometrické řady a dříve dokázané relace $E - I = \Delta$ a $J = \Delta/D$. Podobně můžeme upravit sumu vystupující v dokazované formuli

$$\sum_{j=0}^{p-1} f(x+jh) = [E^0 + E^1 + E^2 + ... + E^{(p-1)}]f(x) = [E^p - I]\Delta^{-1} f(x).$$

Důkaz dokončíme tak, že do posledního vzorce dosadíme za Δ^{-1} z relace

$$\begin{aligned} \frac{1}{2} + \frac{1}{\Delta} &= \frac{1}{2} + \frac{1}{e^{hD} - 1} = \frac{1}{2}\,\frac{e^{hD/2} + e^{-hD/2}}{e^{hD/2} - e^{-hD/2}} \\ &= \frac{1}{2\tanh\frac{hD}{2}} = \frac{1}{hD} + \tfrac{1}{12}hD - \tfrac{1}{720}h^3D^3 + \tfrac{1}{30240}h^5D^5 + ... \end{aligned}$$

a porovnáme obdržené výrazy pro sumu a integrál. Současně je zřejmé jak dostat hodnoty koeficientů a_k. Jde o koeficienty v Taylorově rozvoji funkce $x/2[\tanh(x/2)]^{-1}$.

Poznámky: Eulerovu-Maclaurinovu formuli můžeme používat k odhadu součtu řady integrálem. Lze tak například získat Stirlingovu aproximaci chování $\ln n!$ pro velká n. My se k ní naopak vrátíme při diskusi metod pro výpočet integrálu pomocí součtu funkčních hodnot na pravidelné síti.

Poslední téma, na které aplikujeme elegantní aparát diferenčních operátorů bude přibližné řešení *diferenciálních rovnic*. Samotnému numerickému řešení diferenciálních rovnic se budeme věnovat později. Uvidíme, že nahrazením derivací přibližnými vzorci v diferenciální rovnici dostaneme diferenční rovnice, jejichž speciálním případem jsou **Lineární diferenční rovnice**. Tato je lineárním vztahem, který umožňuje spočíst funkci $f(x)$ v bodě $x = x_0 + nh$, pokud již funkci známe v několika předchozích bodech $x = x_0 + (n-1)h$, ..., $x_0 + (n-k)h$. Pro stručnost označíme $f_n = f(x_0 + nh)$.

Definice: Lineární diferenční rovnicí nazveme vztah

$$a_k f_{n+k} + ... + a_1 f_{n+1} + a_0 f_n = g_n, \tag{3.8}$$

kde g_n je zadaná funkce n a a_0, ..., a_k jsou reálná čísla (obecně to mohou rovněž být funkce n, ale zde se omezíme na konstanty). Funkce f_n je neznámá (vlastně bychom mohli mluvit o posloupnosti f_n), která musí rovnici splňovat pro $n = 0, 1, 2, ...$ Pokud je $g_n = 0$ mluvíme o *homogenní diferenční rovnici*. Číslo $k > 0$ nazýváme *řádem diferenční rovnice* (požadujeme, aby $a_k \neq 0$).

Řešením lineární diferenční rovnice rozumíme nalezení f jako funkce n. Teorie řešení lineárních diferenčních rovnic je velmi podobná řešení lineárních rovnic diferenciálních. Přitom existence řešení je zřejmá, neboť f_0, ..., f_{k-1} můžeme zvolit libovolně a f_k vypočteme přímo z (3.8) pro $n = 0$, f_{k+1} z (3.8) pro $n = 1$ atd. Z toho je rovněž zřejmé, že řešení je jednoznačné pokud zadáme prvních k členů posloupnosti f_0, f_1, ... f_{k-1}. Z linearity vztahu (3.8) je ihned vidět, že řešení homogenní rovnice je k-dimenzionální lineární vektorový prostor a obecné řešení nehomogenní rovnice je dáno partikulárním řešením s danou pravou stranou plus obecné řešení rovnice homogenní, podobně jako pro lineární rovnice diferenciální. Zbývá zodpovědět otázku jak nalézt k lineárně nezávislých řešení homogenní rovnice a partikulární řešení rovnice nehomogenní.

Nejdříve si povšimněme, že $f_{n+i} = E^i f_n$ a rovnici (3.8) tedy můžeme psát ve tvaru

$$p(E) f_n = g_n,$$

kde $p(\lambda)$ je tzv. *charakteristickým polynomem* rovnice (3.8)

$$p(\lambda) \equiv a_k\lambda^k + ... + a_1\lambda + a_0 = a_k(\lambda - \lambda_1)^{\alpha_1}...(\lambda - \lambda_m)^{\alpha_m}.$$

V poslední rovnici jsme si dovolili kromě definice charakteristického polynomu ještě napsat jeho rozklad na součin kořenových činitelů, kde čísla λ_m jsou tyto kořeny a α_m řády těchto kořenů, přičemž z algebry víme, že

$$\alpha_1 + ... + \alpha_m = k$$

Rovnici (3.8) tedy můžeme přepsat jako

$$p(E)f_n = a_k(E - \lambda_1)^{\alpha_1}...(E - \lambda_m)^{\alpha_m}f_n = g_n.$$

Nyní si stačí uvědomit, že pro libovolné komplexní číslo λ je funkce $f_n = \lambda^n$ vlastní funkcí operátoru posunutí $E\lambda^n = \lambda\lambda^n$ a tedy funkce $f_n = \lambda^n$ řeší diferenční rovnici, pokud $\lambda = \lambda_i$ je kořenem charakteristického polynomu. Pokud jsou všechny kořeny jednoduché $\alpha_i = 1$, $\forall i$ máme nalezeno k lineárně nezávislých řešení homogenní rovnice. Pro násobné kořeny si stačí uvědomit, že

$$(E - \lambda)q_N(n)\lambda^n = q_N(n+1)\lambda^{n+1} - q_N(n)\lambda^{n+1} = \tilde{q}_{N-1}(n)\lambda^{n+1},$$

kde q_N respektive $\tilde{q}_{N-1} = \Delta q_N$ jsou nějaké polynomy stupně N a $N-1$ a tedy platí

$$(E - \lambda_i)^{\alpha_i}q_N(n)\lambda^n = 0,$$

pro libovolný polynom $q_N(n)$ stupně menšího než α_i. Jako k lineárně nezávislých řešení rovnice (3.8) lze tedy volit například λ_i^n, $n\lambda_i^n$, $n^2\lambda_i^n$, ... $n^{\alpha_i-1}\lambda_i^n$, $\forall i$.

Poznámka: řešení rovnice s reálnými koeficienty lze volit reálná. Pokud vyjdou komplexní kořeny charakteristického polynomu, vyskytují se v párech navzájem komplexně sdružených kořenů. Součet a rozdíl (vydělený i) příslušných dvou řešení je už reálný.

Příklad: (Fibonacciho posloupnost) Vyřešme homogenní diferenční rovnici

$$f_{n+1} = f_n + f_{n-1},$$

s počáteční podmínkou $f_0 = 1$, $f_1 = 1$. Výsledkem je známá Fibonacciho posloupnost (množící se králící, kolika způsoby lze vyjít schody mohu-li udělat krok přes jeden či dva schody, počet spirál v květech slunečnice, ananasu atd.) 1, 1, 2, 3, 5, 8, 13, 21, ...

Řešení: Charakteristickým polynomem rovnice je $p(\lambda) = \lambda^2 - \lambda - 1$ jehož kořeny jsou $\lambda_{1,2} = (1 \pm \sqrt{5})/2$ a tedy obecné řešení je $f_n = A\lambda_1^n + B\lambda_2^n$. Konstanty A, B najdeme z počáteční

$$\begin{aligned} f_0 &= 1 = A + B, \\ f_1 &= 1 = A\lambda_1 + B\lambda_2 = (A + B + \sqrt{5}(A - B))/2, \end{aligned}$$

tj. $A = \frac{\sqrt{5}+1}{2\sqrt{5}}$, $B = \frac{\sqrt{5}-1}{2\sqrt{5}}$ neboli

$$f_n = \frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^{n+1} - \left(\frac{1-\sqrt{5}}{2}\right)^{n+1}\right].$$

To je pozoruhodný výsledek. Všechny členy Fibonacciho posloupnosti musí být celá čísla, a tento vzorec také celá čísla dává pro každé $n \geq 0$ i když to na první pohled není vidět.

Nyní si ukážeme způsob jak najít *partikulární řešení* alespoň pro speciální volbu g_n. Ukažme si to na příkladu rovnice

$$f_{n+2} - f_{n+1} - f_n = (E^2 - E - I)f_n = n. \tag{3.9}$$

Její formální řešení můžeme psát jako

$$\begin{aligned} f_n &= (E^2 - E - I)^{-1}n = [(I+\Delta)^2 - (I+\Delta) - I]^{-1}n = -[I - \Delta - \Delta^2]^{-1}n, \\ &= -[I+\Delta]n = -En = -(n+1), \end{aligned}$$

přičemž v druhém řádku jsme udělali Taylorův rozvoj v Δ do prvního řádu, neboť vyšší členy vymizí při aplikaci na polynom prvního řádu. Podobný postup lze uplatnit i pro jiné diferenční rovnice a pro pravou stranu ve tvaru libovolného polynomu v proměnné n. Obecné řešení rovnice (3.9) tedy je

$$f_n = A\left(\frac{1+\sqrt{5}}{2}\right)^n + B\left(\frac{1-\sqrt{5}}{2}\right)^n - (n+1).$$

Ovšem konstanty A a B musíme určit znovu. Pro počáteční podmínku $f_0 = f_1 = 1$ dostaneme

$$f_n = \frac{\sqrt{5}+2}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^n + \frac{\sqrt{5}-2}{\sqrt{5}}\left(\frac{1-\sqrt{5}}{2}\right)^n - (n+1).$$

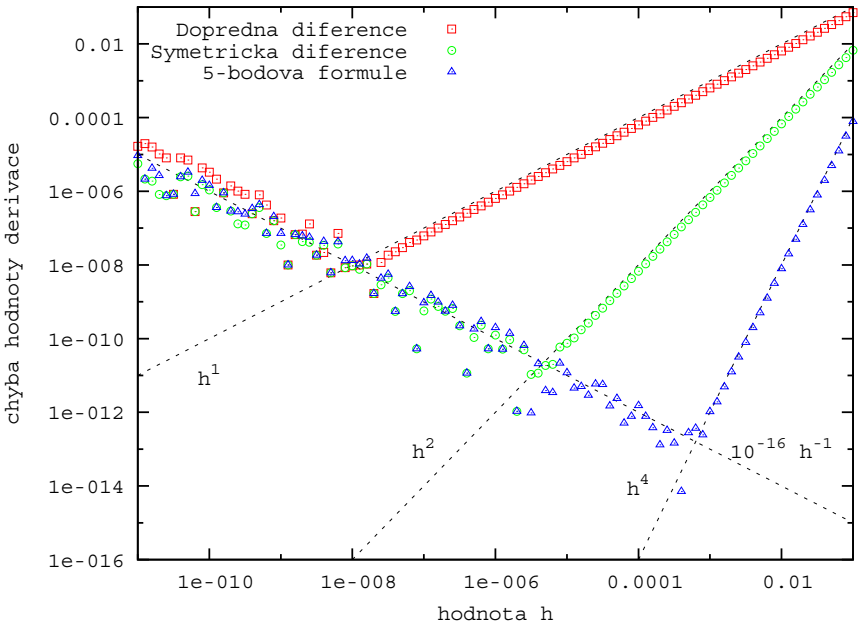
3.4 Numerická derivace

Tematiky numerického derivování jsme se letmo dotkli v minulém odstavci. Podařilo se nám odvodit některé vzorce pro numerickou derivaci funkce tabelované na rovnoměrné síti. Zde ještě odvodíme některé vzorce klasickým způsobem s použitím Taylorova rozvoje. Budeme proto předpokládat, že funkce $f(x)$ je dostatečněkrát spojitě diferencovatelná a dále si označíme f_i hodnotu v bodě $f(x+ih)$, kde h je krok diskretizační sítě. Uveďme nejdříve několik nejpoužívanějších vzorců

$$\begin{aligned}
f'(x) &= [f_1 - f_0]/h + O(h), && (\ -\tfrac{1}{2}hf''\ ), && \text{(D1)}\\
&= [f_0 - f_{-1}]/h + O(h), && (\ \tfrac{1}{2}hf''\ ), && \text{(D2)}\\
&= [f_1 - f_{-1}]/2h + O(h^2), && (-\tfrac{1}{6}h^2f'''), && \text{(D3)}\\
&= [-f_2 + 4f_1 - 3f_0]/2h + O(h^2), && (\ \tfrac{1}{3}h^2f'''\ ), && \text{(D4)}\\
&= [3f_0 - 4f_{-1} + f_{-2}]/2h + O(h^2), && (\ \tfrac{1}{3}h^2f'''\ ), && \text{(D5)}\\
&= [-f_2 + 8f_1 - 8f_{-1} + f_{-2}]/12h + O(h^4), && (\ \tfrac{1}{30}h^4f^v\ ), && \text{(D6)}
\end{aligned}$$

přičemž formule (D1) a (D4) jsou přímým důsledkem formule (3.3) použité na interpolační polynom prvního respektive druhého řádu a vzorce (D2) a (D5) jsou jejich analogie pro zpětnou diferenci. Symetrický vzorec (D3) plyne z (3.4) po aplikaci na interpolační polynom druhého řádu. Povšimněte si výrazů v závorce. Ty upřesňují nejnižší člen v asymptotickém rozvoji pro $h \to 0$. Ten snadno naleznete dosazením Taylorových rozvojů

$$\begin{aligned}
f_{-2} &= f_0 - 2hf' + 2h^2f'' - \tfrac{8}{6}h^3f''' + \tfrac{16}{24}h^4f^{iv} + O(h^5),\\
f_{-1} &= f_0 - hf' + \tfrac{1}{2}h^2f'' - \tfrac{1}{6}h^3f''' + \tfrac{1}{24}h^4f^{iv} + O(h^5),\\
f_1 &= f_0 + hf' + \tfrac{1}{2}h^2f'' + \tfrac{1}{6}h^3f''' + \tfrac{1}{24}h^4f^{iv} + O(h^5),\\
f_2 &= f_0 + 2hf' + 2h^2f'' + \tfrac{8}{6}h^3f''' + \tfrac{16}{24}h^4f^{iv} + O(h^5).
\end{aligned}$$



Obrázek 3.1: Vliv zaokrouhlovací chyby a chyby aproximace při výpočtu derivace z konečných diferencí funkce (vzorce D1,D3,D6).

do vzorců pro derivaci. Pomocí těchto rozvojů lze také dospět ke vzorcům (D1)-(D6) alternativním (pracnějším) způsobem — metodou neurčitých koeficientů. Například k odvození vzorce (D4) potřebujeme najít koeficienty a_0, a_1 a a_2 v lineární kombinaci $DF = a_0 f_0 + a_1 f_1 + a_2 f_2$, tak aby výsledný Taylorův rozvoj výrazu DF obsahoval koeficient 0 u mocniny h^0, koeficient 1 u mocniny h^1 a koeficient 0 u mocniny h^2. To vede na soustavu lineárních rovnic určující koeficienty a_i. Podobně k odvození (D6) hledáme koeficienty u pěti funkčních hodnot f_{-2}, f_{-1}, f_0, f_1, f_2 (koeficient u f_0 vychází u výsledného výrazu nulový).

Ke vzorci (D6) lze rovněž dospět metodou *Richardsonovy extrapolace* ze vzorce (D3). Podobně jako při odvození Aitkenova Δ^2-vzorce využijeme znalosti asymptotického chování chyby. Definujeme

$$Df(h) \equiv \frac{1}{2h}[f_1 - f_{-1}] = f' + Ch^2 + O(h^4)$$

a tedy

$$Df(2h) \equiv \frac{1}{4h}[f_2 - f_{-2}] = f' + 4Ch^2 + O(h^4).$$

Z toho vidíme, že vezmeme-li čtyřnásobek prvního vzorce a odečteme vzorec druhý, první část chybového členu se vyruší, tj.

$$\frac{4Df(h) - Df(h)}{3} = 4\frac{f_1 - f_{-1}}{6h} - \frac{f_2 - f_{-2}}{12h} = f' + O(h^4),$$

což je až na přeuspořádání členů vzorec (D6).

Optimální volba hodnoty h pro numerický výpočet derivace.

Uvedené vzorce lze využít ke skutečnému numerickému nalezení derivace pokud máme k dispozici proceduru, která vrací funkční hodnotu $f(x)$ v daném bodě x, nebo pokud máme funkci f tabelovanou s rovnoměrným krokem h. K ujasnění vlivu volby délky kroku h na výslednou hodnotu derivace použijeme některé z uvedených vzorců pro výpočet derivace funkce

$f(x) = \exp(\cos x)$ v bodě $x = 1$. Funkci umíme derivovat přesně, takže můžeme například vypočítat chybu $|f' - Df(h)|$ aproximace hodnoty $f'(x)$ pomocí vzorce (D1) v závislosti na velikosti h. Výsledek je ukázán na obrázku 3.1 (červené čtverečky). Výsledný obrázek je daný soupeřením dvou trendů:

- *Chyba aproximace:* Pro větší hodnoty h chyba roste lineárně s h v souladu s předpokládanou teoretickou chybou $hf''/2$ aproximace derivace vzorcem (D1).
- *Zaokrouhlovací chyba:* Pro opravdu malé hodnoty h převládne zaokrouhlovací chyba. Pokud předpokládáme, že relativní chyba výpočtu funkční hodnoty je omezená strojovým ϵ, musí numerická funkční hodnota $f(x)$ ležet v intervalu $f(x)(1 \pm \epsilon)$. Absolutní chybu výsledku vzorce $Df(h)$ můžeme tedy shora odhadnout hodnotou $2\epsilon|f|/h$.

Zaokrouhlovací chyba se tedy zmenšuje s rostoucím h, zatímco chyba aproximace se s rostoucím h naopak zvětšuje. Nejmenší celkové chyby dosáhneme pro takové h_0, pro něž budou obě chyby zhruba stejně velké, tj. platí

$$\frac{h_0|f''|}{2} \simeq \frac{2\epsilon|f|}{h_0}, \qquad \text{neboli} \qquad h_0 \simeq 2\sqrt{\frac{|f|}{|f''|}}\sqrt{\epsilon}$$

Faktor $D = 2\sqrt{|f/f''|}$ má stejný rozměr jako x a h a můžeme jej chápat jako charakteristickou škálu na níž se podstatně mění funkce f. Přesnost samotné formule charakterizuje faktor $\sqrt{\epsilon}$. V případě vzorce (D1) při použití dvojité přesnosti vychází optimální $h_0 = \sqrt{\epsilon} \simeq 10^{-8}$ ve shodě s minimem červené křivky na obrázku 3.1. Na stejném obrázku jsou ukázány též hodnoty chyby pro vzorce D3(zelená kolečka) a D6(modré trojúhelníčky). Vidíme, že zatímco zaokrouhlovací chyba v levé části grafu je pro všechny metody zhruba stejná, chyba aproximace se pro jednotlivé metody dramaticky liší. Její chování je řádově $f'''h^2$ respektive $f^v h^4$ pro výrazy D3 a D6. Zopakováním předchozí analýzy pro tyto případy dostaneme optimální volbu kroku řádově $h_0 \simeq \sqrt[3]{\epsilon} \simeq 10^{-5}$ a $h_0 \simeq \sqrt[5]{\epsilon} \simeq 10^{-3}$ ve shodě s obrázkem. Z obrázku 3.1 rovněž vidíme, že řád metody (mocnina h v odhadu chyby) určuje nejen to jaké optimální $h = h_0$ zvolit pro největší přesnost výsledku, ale určuje přesnost samou. Pro vzorec vyššího řádu obecně můžeme dostat vyšší přesnost výsledku. Například pro právě analyzované vzorce (D1), (D3) respektive (D6) je nejmenší dosažitelná chyba řádově $h_0^1 = \sqrt{\epsilon} \simeq 10^{-8}$, $h_0^2 = \epsilon^{2/3} \simeq 3 \times 10^{-11}$ a $h_0^4 = \epsilon^{4/5} \simeq 3 \times 10^{-13}$.

Pro úplnost uveďme ještě pár vzorců pro vyšší derivace

$$\begin{array}{ccc} f'' & = & \dfrac{f_1 - 2f_0 + f_{-1}}{h^2} + O(h^2), \\ \ldots & & \ldots \end{array}$$

Analýzu výběru optimální hodnoty lze provést stejně jako pro první derivaci, jen zaokrouhlovací chyba pro $k-$tou derivaci je úměrná h^{-k}.

3.5 Numerická integrace

V této kapitole se budeme věnovat numerickému výpočtu určitého integrálu funkce

$$I[f(x)] \equiv \int_a^b f(x)dx. \tag{3.10}$$

Přitom bychom integrál rádi nahradili konečnou sumou (tzv. *kvadraturním vzorcem*)

$$I[f(x)] \simeq I_N[f(x)] \equiv \sum_{j=0}^{N} w_j f(x_j), \tag{3.11}$$

kde N je počet intervalů délky $h = |b-a|/N$ na něž jsme rozdělili integrační oblast. O bodech $x_i = a + ih$ v nichž vyčíslujeme integrovanou funkci $f(x)$ se v tomto kontextu často mluví jako o *uzlech* kvadraturního vzorce a o koeficientech w_i jako o *vahách*. Forma vzorce (3.11) je vlastně zobecněním Riemanovy definice integrálu, který dostaneme pro $h \to 0$. Většina kvadraturních vzorců vychází z toho, že funkce je dostatečně hladká a lze ji aproximovat dobře polynomem. Příkladem je lichoběžníkové (3.5) a Simpsonovo pravidlo (3.6). Ty jsme odvodili tak, že jsme vyjádřili operátor integrace pomocí rozvoje v operátoru diference a ořezáním do prvního respektive druhého řádu v Δ jsme zaručili, že vzorec integruje přesně polynom prvního respektive druhého řádu. Protože vzorce současně obsahují funkční hodnoty f_0, f_1 respektive f_0, f_1, f_2 platí pro obecnou funkci $f(x)$, že tyto vzorce integrují přesně interpolační polynom prvního respektive druhého řádu, proložený těmito hodnotami. Zobecněním těchto myšlenek je pojem Newtonových-Cotesových vzorců.

Definice: Řekneme, že vzorec (3.11) je **Newtonův-Cotesův**, pokud $I_N(x^n) = I[x^n]$ pro všechna $n = 0, 1, 2, ..., N$.

Tato definice požaduje, aby kvadraturní vzorec splňoval $N+1$ podmínek. Přitom vzorec obsahuje $N+1$ neznámých konstant w_i, které jsou těmito podmínkami jednoznačně určeny. Příslušnou soustavu rovnic si za chvíli napíšeme v trochu obecnějším případě při diskusi Gaussovy kvadratury.

Při praktickém použití Newtonových-Cotesových vzorců nemůžeme příliš zvyšovat řád N. To nahlédneme z toho, že tyto vzorce vlastně integrují interpolační polynom. Z diskuse interpolace funkcí víme, že interpolační polynomy mají problémy vystihnout funkce, které mají nespojitosti v některé derivaci, ale i pro zcela hladkou funkci se může interpolační polynom vysokého řádu značně lišit od interpolované funkce v intervalech mezi uzlovými body x_i. Jeden ze způsobů nápravy je vzdát se jednoduché sítě pravidelně rozmístěných bodů x_i. Jak jsme viděli v kapitole o interpolaci, vhodně zvolená síť vede k dobrým vlastnostem interpolace dokonce i pro nehladké funkce. Tuto strategii rovněž rozvineme detailněji v následujícím odstavci o Gaussově kvadratuře.

Druhou možností nápravy je rozdělit integrační oblast na mnoho malých intervalů a na nich opakované použít Newtonových-Cotesových vzorců nízkého řádu. To vede k definici *složených Newtonových-Cotesových* vzorců. Uveďme si alespoň dva nejnižší. Rozdělením integrační oblasti $\langle x_0, x_N\rangle$ na intervaly $\langle x_0, x_1\rangle$, $\langle x_1, x_2\rangle$, $\langle x_2, x_3\rangle$, ... a použitím lichoběžníkového pravidla (3.5) na každém z nich dostaneme *(rozšířené) lichoběžníhové pravidlo*

$$I_N^0[f(x)] = h\left[\tfrac{1}{2}f_0 + f_1 + f_2 + ... + f_{N-1} + \tfrac{1}{2}f_N\right]. \tag{3.12}$$

Rozdělení integrační oblasti $\langle x_0, x_N\rangle$ na intervaly $\langle x_0, x_2\rangle$, $\langle x_2, x_4\rangle$, $\langle x_4, x_6\rangle$, ... (budeme předpokládat, že N je sudé) a použití Simpsonova pravidla (3.6) na každém z nich vede na *(rozšířené) Simpsonovo pravidlo*

$$I_N^1[f(x)] = \frac{h}{3}\left[f_0 + 4f_1 + 2f_2 + 4f_3 + 2f_4 + ... + 4f_{N-1} + f_N\right]. \tag{3.13}$$

Nyní bychom se mohli pokoušet pokračovat a konstruovat Newtonovy-Cotesovy vzorce vyšších řádů a jejich složené varianty. Ukážeme si elegantní metodu jak toho dosáhnout bez toho

abychom konstruovali každý vzorec zvlášť. Klíčem k tomuto postupu je porozumění diskretizační chybě rozšířeného lichoběžníkového pravidlaTu udává Eulerova-Maclaurinova sumační formule

$$I_N^0[f] = I[f] + a_1[f'_N - f'_0]h^2 + a_2[f_N^{(3)} - f_0^{(3)}]h^4 + ...$$

Zdůrazněme ještě, že nejde o konvergentní, ale o asymptotický rozvoj, tj. například po zahrnutí členu úměrného rozdílu prvních derivací je zbytek $O(h^2)$, po zahrnutí dalšího členu je zbytek $O(h^4)$ atd. V kapitole 2 při odvození Aitkenova Δ^2-vzorce jsme viděli, že znalost chování chyby umožňuje konstrukci přesnějšího výsledku tzv. metodou *Richarsonovy extrapolace*. Nyní si uvědomíme, že

$$\begin{aligned} I_N^0[f] &= I[f] + ch^2 + O(h^4), \\ I_{2N}^0[f] &= I[f] + c\tfrac{1}{4}h^2 + O(h^4). \end{aligned}$$

Odtud je ihned vidět, že

$$I_{2N}^1[f] \equiv \frac{4I_{2N}^0 - I_N^0}{3} = I[f] + O(h^4).$$

Přitom se snadno přesvědčíte, že výsledné I_N^1 je totožné výsledku Simpsonova pravidla (3.13). Nyní můžeme pokračovat a pokusit se zkonstruovat $I^2[f]$ odstraněním chyby řádu h^4, $I^3[f]$ odstraněním $O(h^6)$ atd. Obecně dostáváme

$$I_{2N}^k[f] \equiv \frac{4^k I_{2N}^{k-1} - I_N^{k-1}}{4^k - 1} = I[f] + O(h^{2(k+1)}). \tag{3.14}$$

Ještě si uvědomíme, že pro výpočet I_{2N}^0 můžeme využít informace obsažené v I_N^0

$$I_{2N}^0[f] = \tfrac{1}{2}I_N^0[f] + h\sum_{j=1}^{N} f(x_{2j+1}), \tag{3.15}$$

kde faktor 1/2 pochází z toho, že pro výpočet $I_N^0[f]$ používáme dvakrát větší krok než pro $I_{2N}^0[f]$. Nyní můžeme formulovat *Rombergův algoritmus*, který spočívá v postupném výpočtu tabulky

$$\begin{array}{llll} I_1^0[f] & & & \\ I_2^0[f] & I_2^1[f] & & \\ I_4^0[f] & I_4^1[f] & I_4^2[f] & \\ \vdots & \vdots & ... & \ddots \\ I_{2^k}^0[f] & I_{2^k}^1[f] & ... & I_{2^k}^k[f] \end{array}$$

Tabulku generujeme postupně po řádcích, přičemž vždy další číslo v řádku nalezneme z předchozího pomocí vzorce (3.14) a první číslo v každém sloupci vypočteme pomocí vzorce (3.15) přičemž vždy zmenšujeme krok h na polovinu. Startovací hodnota $I_1^0 = \frac{b-a}{2}[f(a) + f(b)]$ je prostě (srovnejte se vzorcem (3.5)) tj. počáteční $h = b - a$.

Poznámky:

- Pro funkce, které mají charakteristické škály (např. perioda, šířka peaku) hodně menší než velikost integrační oblasti $|b-a|$ může být vhodné odstartovat Rombergův algoritmus od hodnoty I_N^0 místo I_1^0 přičemž N je voleno tak, aby počáteční krok $|b-a|/N$ byl srovnatelný s charakteristickou škálou.

- Nezapomeňte, že Rombergův algoritmus vychází z Eulerova-Maclaurinova vzorce. Ten byl odvozen pro funkce dobře aproximované polynomy. Obecně $k-$tý sloupec I^k Rombergovy tabulky bude dobře aproximovat integrál $I[f]$ pokud funkce f má spojitých $2k-1$ derivací. Pokud funkce není dostatečně spojitá, musíme dalšího zpřesňování odhadu I_N^k dosahovat zvětšováním N a ne k.

- Platnost Eulerova-Maclaurinova vzorce je třeba mít v paměti i z následujícího důvodu. Může se stát, že derivace funkce v krajních bodech jsou z nějakého důvodu nulové nebo shodné (například při integraci periodické funkce přes periodu). Potom jeden nebo několik členů asymptotického rozvoje je nulových a použití Richardsonovy extrapolace pro jejich odstranění je nesprávné.

- Právě případ integrace hladké periodické funkce přes její periodu zasluhuje zvláštní pozornost. Tam právě všechny členy odhadu chyby pomocí Eulerova-Maclaurinovy formule vymizí. To neznamená, že chyba je nulová, ale spíše, že chyba ubývá se zmenšujícím h rychleji než jakákoli mocnina h^m, tj. např. exponenciálně. V tomto případě nemá smysl používat Rombergův algoritmus, ale už samotné odhady integrálu pomocí lichoběžníkového pravidla konvergují s rostoucím N velmi rychle ke správné hodnotě. To je velmi důležité pro výpočet diskrétní fourierovy tranformace.

- Dosud jsme se věnovali analýze diskretizační chyby kvadraturních formulí. Viděli jsme, že diskretizační chyba složeného lichoběžníkového pravidla je řádu $O(h^2)$, diskretizační chyba složeného Simpsonova řádu $O(h^4)$ a Rombergův algoritmus umožňuje konstruovat systematicky odhady s přesností vyšších řádů. Při analýze numerického výpočtu derivace jsme viděli, že finální dosaženou přesnost podstatně ovlivňuje rovněž zaokrouhlovací chyba. Při výpočtu integrálů, většinou zaoktrouhlovací chyba výsledky podstatně neovlivňuje, díky násobení h v kvadraturních vzorcích a pro rozumné funkce není obtížné dosáhnout výsledné přesnosti jen několikrát horší než strojové epsilon (vyjímkou může být například integrace oscilujících funkcí).

Poznámky o kvadratuře pro funkce s nespojitostmi singularitami atd.

Doposud jsme si ukázali jak lze velice efektivně integrovat přes konečný interval funkci, která je na tomto intervalu dostatečně hladká v celém intervalu. Pokud má funkce nebo její derivace nespojitosti nebo singularity, nebo pokud chceme počítat nevlastní integrály, nejsou Rombergova metoda vhodná pro výpočet integrálu. V nejjednodušším případě, kdy k problémům dochází v konečném počtu bodů, lze integrační interval rozřezat na několik kusů a Rombergovu kvadraturu použít po částech. Přitom se může stát, že funkce není definovaná v krajních bodech intervalu. Potom lze uvažovat o použití tzv. otevřených vzorců. Například ...

Jinou metodou odstranění problémů je použití vhodné úpravy integrálu, kterých chceme spočíst. Ukážeme si to na příkladě výpočtu Hilbertovy transformace. Hilbertova transformace funkce $f(x)$ je definována následujícím integrálem

$$Hf(y) \equiv v.p. \int \frac{f(x)}{y-x} \mathrm{d}x,$$

kde $v.p.$ označuje hlavní hodnotu integrálu. Integrand má singularitu v bodě $x = y$. Tato singularita se dá odstranit, pokud najdeme funkci $g(x)$, jejíž Hilbertovu transformaci $Hg(y)$ známe, pokud je současně funkce $\tilde{f}(x) \equiv f(x)/g(x)$ hladká, neboť můžeme psát (k funkci $\tilde{f}(x)$

v integrandu jsme přičetli a odečetli $\tilde{f}(y)$)

$$Hf(y) = v.p. \int \frac{\tilde{f}(x)g(x)}{y-x}\mathrm{d}x = \int \frac{[\tilde{f}(x)-\tilde{f}(y)]g(x)}{y-x}\mathrm{d}x + \tilde{f}(y)Hg(y).$$

Integrand v posledním integrálu je již hladkou funkcí, neboť bod $x = y$ ve výrazu

$$[\tilde{f}(x)-\tilde{f}(y)]/(x-y)$$

představuje odstranitelnou singularitu.

Toto byl jen jeden příklad odstranění singularity úpravou integrálu. Obecně je v tomto případě potřebná schopnost jisté matematické tvořivosti. Kromě toho existují rozsáhlé knihovny pro integraci výrazů speciálního tvaru jejichž rozbor je nad rámec této přednášky (viz například ...). My si uvedeme ještě jeden obecnější přístup zahrnutí singularit a nespojitostí do numerické kvadratury, který spočívá v zavedení váhové funkce. Nejdříve si pro tento případ zobecníme Newtonovy-Cotesovy vzorce a poté se seznámíme s pojmy:

3.5.1 Gaussova kvadratura a ortogonální polynomy

V této kapitole budeme následovat pana Gausse a dovedeme kvadraturní vzorce k dokonalosti. Nejprve pár definic.

Uvažujeme následující vzorec pro výpočet integrálu

$$I[f(x)] \equiv \int_a^b f(x)w(x)dx \simeq \sum_{j=1}^{N} w_j f(x_j) \equiv I_N[f(x)]. \tag{3.16}$$

Tento vzorec se podobá Newtonovým-Cotesovým vzorcům, kromě toho, že jsme si rozdělili integrand na dvě části. O funkci $w(x)$ budeme předpokládat, že je skoro všude kladná na intervalu $\langle a, b\rangle$, a že integrál $I[x^n]$ konverguje pro všechna n a budeme o ní mluvit jako o *váhové funkci*. Výraz $I_N[f(x)]$ nazveme kvadraturním vzorcem. Podobně jako v případě Newtonových-Cotesových vzorců budeme hledat koeficienty w_j tak, aby kvadraturní vzorec byl co nejpřesnější a budeme chtít, aby vzorec byl přesný, kdykoli je funkce $f(x)$ polynomem nízkého stupně. $I[f]$ i $I_N[f]$ jsou lineárními funkcionály a tudíž platí, že je-li vzorec (3.16) přesný na nějaké množině funkcí, je přesný také na lineárním obalu této množiny. Řekneme, že vzorec $I_N[f]$ je Newtonův-Cotesův, pokud $I_N[x^n] = I[x^n] \equiv \mu_n$, pro všechna $n = 0, 1, ..., N-1$. To vede na následující soustavu lineárních rovnic pro koeficienty w_j

$$\mathrm{Vw} = \mu$$

kde $\mathbf{w}$ a μ jsou sloupcové vektory tvořené koeficienty w_j a integrály $\mu_m \equiv I[x^n]$ (kterým říkáme momenty míry $w(x)\mathrm{d}x$). S *Vandermode*ho maticí Vw jsme se již setkali v kapitole o Lagrangeově interpolaci a víme tedy, že pro navzájem různé body x_j, je tato matice regulární a tedy existuje právě jedna sada vah w_j, tak že vzorec $I_N[f]$ je Newtonův-Cotesův. Počítáním na prstech zjistíme, že počet neznámých koeficientů w_j odpovídá dimenzi prostoru polynomů, pro které funguje kvadraturní vzorec přesně. Body x_j figurují v kvadraturním vzorci jako pevné parametry a v případě klasických Newtonových-Cotesových vzorců jsou voleny jako pravidelná síť bodů s rozestupem $h > 0$. Pokud se budeme na čísla x_j budeme dívat jako na proměnné budeme moci zvětšit dimenzi prostoru polynomů, které integruje kvadraturní vzorec přesně.

Definice: Řekneme, že kvadraturní vzorec $I_N[f]$ je Gaussův, pokud $I_N[x^n] = I[x^n]$, pro všechna $n = 0, 1, ..., 2N-1$.

Zatímco nalezení parametrů w_j, bylo přímočaré a vedlo na lineární problém, rovnice pro x_j tvoří komplikovaný nelineární systém. Pro jejich nalezení budeme muset udělat odbočku.

Definice (*Ortogonální polynomy*): Řekneme, že posloupnost polynomů $\{p_0(x), p_1(x), p_2(x), ...\}$ tvoří ortogonální systém polynomů, pokud $p_n(x)$ je polynom stupně n a $I[p_n(x)q(x)] = 0$ pro každý polynom stupně nejvýše $n-1$.

Speciálně pro ortogonální polynomy platí $I(p_i(x)p_j(x)) = A_i\delta_{ij}$, kde $A_i > 0$. Zadané $I[f(x)]$ vlastně definuje skalární součin

$$(p(x)|q(x)) \equiv I(p(x)q(x))$$

na prostoru polynomů a posloupnost $p_0(x)$, $p_1(x)$, $p_2(x)$ lze konstruovat postupnou ortogonalizací množiny 1, x, x^2, ... Z toho je zřejmé, že ortogonální polynomy jsou pro danou váhovou funkci $w(x)$ určeny jednoznačně, až na normalizační konstantu A_i.

Kdy je kvadratura $I_N[f(x)]$ Gaussova?
Předpokládejme, že váhy w_j v I_N jsou zvoleny tak, aby kvadraturní vzorec byl Newtonův-Cotesův. Libovolný polynom $p(x)$, stupně nejvýše $2N-1$ můžeme napsat jako $p(x) = q(x)p_N(x)+r(x)$, kde $q(x)$ a $r(x)$ jsou podíl a zbytek po dělení polynomu $p(x)$ N-tým prvkem $p_n(x)$ ze systému ortogonálních polynomů příslušných funkcionálu $I[f(x)]$. Jsou to tedy polynomy stupně nejvýše $N-1$ a platí

$$I[p(x)] = I[q(x)p_N(x)] + I[r(x)] = I_N[r(x)],$$

kde jsme využili nejdříve toho, že $p_N(x)$ je ortogonální polynom a tedy $I[q(x)p_n(x)] = 0$ a dále toho, že vzorec je Newtonův-Cotesův a tedy $I[r(x)] = I_N[r(x)]$. Aby, byl vzorec I_N Gaussův, musí platit

$$I[p] = I_N[p] = \sum_{J=1}^{N} w_j q(x_j)p_N(x_j)] + I_N[r(x)]$$

pro libovolnou volbu polynomu $q(x)$. To je možno splnit jedině tehdy, pokud x_1, x_2, ..., x_N jsou kořeny polynomu $p_n(x)$.

Dospíváme k závěru: *kvadraturní vzorec* (3.16) *je Gaussův, právě když uzly x_j jsou kořeny ortogonálního polynomu $p_N(x)$*. Ze základní věty algebry, plyne, že těchto kořenů je právě N. Později si řekneme více o jejich hledání a také si dokážeme, že jsou všechny navzájem různé (tj. $p_N(x)$ má jen jednoduché kořeny).

Nejčastěji se vyskytující Gaussovy kvadratury jsou shrnuty v následující tabulce

Gauss-Legendre:	$\langle -1, 1\rangle$ s váhou $w(x) = 1$,
Gauss-Chebychev:	$\langle -1, 1\rangle$ s váhou $w(x) = \frac{1}{\sqrt{1-x^2}}$,
Gauss-Jacobi:	$\langle -1, 1\rangle$ s váhou $w(x) = (1-x)^\alpha(1+x)^\beta$,
Gauss-Laguere:	$\langle 0, \infty)$ s váhou $w(x) = x^\alpha e^{-x}$,
Gauss-Hermite:	$(-\infty, \infty)$ s váhou $w(x) = e^{-x^2}$.

První tři kvadratury odpovídají integraci v konečných mezích, kterou lze lineární substitucí $z = ax + b$ převést vždy na interval $\langle -1, 1\rangle$. Jednotlivé možnosti odpovídají různým singularitám v krajních bodech (pokud se singularita nachází ve vnitřním bodě můžeme vždy rozdělit integraci na dvě části). Gaussova-Laguerova kvadratura umožňuje provádět numerickou integraci s jednou, a Gaussova-Hemitova se dvěma nevlastními mezemi. Podrobnosti o nalezení

uzlových a váhových bodů, lze nalézt v Numerických receptech [], včetně podprogramů v různých programovacích jazycích a některých vlastnostech příslušných ortogonálních polynomů. Chybové vzorce pro některé kvadratury naleznete například v [].

Někdy se setkáme se situací, kdy musíme konstruovat Gaussův kvadraturní vzorec sami pro nějakou nestandardní váhovou funkci $w(x)$. Přímočará konstrukce posloupnosti ortogonálních polynomů $p(x)$ a následné nalezení kořenů nepředstavuje stabilní algoritmus, neboť úloha nalezení kořenů polynomu ze znalosti jeho koeficientů, není dobře podmíněná. V následujícím odstavci převedeme úlohu pro nalezení uzlových bodů x_i na nalezení vlastních čísel simetrické tridiagonální matice, což je dobře podmíněná úloha na jejíž řešení existují efektivní algoritmy (viz kapitola věnující se numerické lineární algebře).

Nejdříve nalezneme elegantní způsob konstrukce posloupnosti ortogonálních polynomů. Podívejme se na polynom $xp_n(x)$. To je polynom řádu $n+1$ a musí jít tudíž napsat jako lineární kombinace prvních $n+1$ polynomů ortogonálního systému

$$xp_n(x) = \sum_{k=0}^{n+1} a_k p_k(x),$$

kde $a_k = (p_k|xp_n)/(p_k|p_k)$. Přitom z definice ortogonálních polynomů víme, že skalární součin $(p_k|xp_n)$ může být nenulový jen pro $k = n-1, n, n+1$. Výše uvedená rovnice tudíž představuje tří-člennou rekurentní relaci, pomocí níž lze konstruovat postupně posloupnost ortogonálních polynomů. To lze napsat jako následující

Lanczosův algoritmus:
$\beta_{-1} = 0$, $p_{-1}(x) = 0$, $p_0(x) = 1/||1||$
`pro` $n = 0, 1, 2, ..$:

$$\begin{aligned} v(x) &= xp_n(x), \\ \alpha_n &= (p_n|v), \\ v(x) &= v(x) - \beta_{n-1}p_{n-1}(x) - \alpha_n p_n(x), \\ \beta_n &= ||v||, \\ p_{n+1}(x) &= v(x)/\beta_n \end{aligned}$$

Tento algoritmus je pozoruhodný sám osobě, zastavíme se u něj tudíž trochu podrobněji. Lze jej totiž chápat obecněji jako postupnou konstrukci ortonormální baze vektorů p_k, $k = 0, 1, 2, ...$ v níž je předem zadaný operátor A tridiagonální (v našem případě je $A = x$, tj. operátor násobení nezávislou proměnnou x v prostoru polynomů) tj.

$$\begin{aligned} \langle p_n|Ap_n\rangle &= \alpha_n, \\ \langle p_{n+1}|Ap_n\rangle &= \langle p_n|Ap_{n+1}\rangle = \beta_n, \\ \langle p_m|Ap_n\rangle &= 0, \qquad \text{pro } |m-n| > 1. \end{aligned}$$

Přitom počáteční vektor p_0 si obecně můžeme volit libovolně (v případě konstrukce ortogonálních polynomů samozřejmě ne), a ostatní vektory tvoří postupně ortogonální bazi v tzv. Krylovově prostoru, který je dál jako lineární obal vektorů p_0, Ap_0, A^2p_0, ... A^np_0.

Kapitola 4

Numerická integrace diferenciálních rovnic

V této kapitole se budeme zabývat numerickým řešením diferenciálních rovnic. Nejdříve si trochu detailněji vyložíme elegantní teorii lineárních multikrokových metod (podrobnější poučení naleznete v [3]) a potom se ještě zmíníme o metodách typu Runge-Kuta [1]. Tyto první dvě části se věnují řešení diferenciálních rovnic prvního řádu. V posledním odstavci se ještě věnujeme stručně jedné metodě pro přímé řešení některých typů rovnic druhého řádu.

Při zběžném prohlédnutí této kapitoly můžete získat dojem, že existuje nepřeberné množství metod řešení obyčejných diferenciálních rovnic a při praktickém použití se vám může najednou stát, že nevíte kam sáhnout. Tato kapitola neslouží jako praktická příručka, ale spíše ukázka způsobu myšlení, které vede ke konstrukci různých metod a způsobu analýzy konvergence a stability metod. Před praktickým použitím doporučuji sáhnout k numerickým receptům [1]. Obecná rada je použít metody typu Runge-Kutta pro jednodušší úlohy či úlohy nevyžadující extrémní přesnost a multikrokové metody s Richardsonouvou extrapolací pro dosažení přesnosti blížící se strojové přesnosti. Pro problémy komplikované fenoménem silného tlumení je třeba použít implicitní metody (v případě, že problém není výpočetně náročný se lze někdy rovněž uchýlit k metodám typu Runge-Kutta s dostatečně malým integračním krokem). Podstata problému silného tlumení je vyložena na konci kapitoly o linearních multikrokových metodách, ale obecně lze říci, že v případě systému velkého množství svázaných rovnic tento problém téměř jistě nastane.

4.1 Úloha a úvodní poznámky

Nalezněte funkci $u(t) : \langle 0, T\rangle \mapsto \mathbb{C}^d$ splňující rovnici

$$u'(t) \equiv \frac{\mathrm{d}u}{\mathrm{d}t} = f(u(t), t), \tag{4.1}$$

a počáteční podmínku $u(t = 0) = u_0$, kde u_0 je předem zadaný vektor a $f(u, t)$ předem zadaná funkce $f : \mathbb{C}^d \times \langle 0, T\rangle \mapsto \mathbb{C}^d$.

Poznámky:

- Povšimněte si, že úlohu jsme formulovali poměrně obecně. Předně pracujeme s vektorem řešení $u(t) \equiv (u_1(t), ..., u_d(t))$, tj. řešíme ne jednu rovnici, ale soustavu rovnic

$$\begin{aligned} u_1'(t) &= f_1(u_1(r), ..., u_d(t), t), \\ &\vdots \\ u_d'(t) &= f_d(u_1(r), ..., u_d(t), t). \end{aligned}$$

  Navíc obecně uvažujeme komplexní funkce reálné proměnné t.

- Rovnice řešíme na intervalu $\langle 0, T \rangle$. To nečiní žádnou újmu na obecnosti, neboť substitucí, lze počátek řešení vždy posunout do $t = 0$.

- V rovnici vystupuje jen první derivace. Z úvodního kurzu matematické analýzy víte, že libovolnou diferenciální rovnici vyššího řádu lze převést na soustavu rovnic prvního řádu zavedením nových neznámých funkcí $u_1(t) = u'(t)$, $u_2(t) = u''(t) = u_1'(t)$, ...

- Počáteční podmínka rovněž představuje ne jednu, ale d podmínek. Kromě počáteční úlohy, lze rovněž řešit, tzv okrajovou úlohu, v níž se požaduje splnění d_1 podmínek v bodě $t = 0$ a d_2 podmínek v bodě $t = T$, přičemž $d_1 + d_2 = d$. Okrajovou úlohou se budeme zabývat až v dalším semestru, neboť metody k jejímu řešení úzce souvisí s metodami pro řešení parciálních diferenciálních rovnic. Zde se budeme věnovat výhradně počáteční úloze.

Několik příkladů konstrukce metod řešení

Při řešení počáteční úlohy pro obyčejné diferenciální rovnice budeme vždy konstruovat řešení na rovnoměrné síti $t_n = nh$ v nezávislé proměnné t, přičemž h je krokem sítě. Řešení budeme reprezentovat hodnotami v_n, které by měly v ideálním případě být rovny správnému řešení $u_n = u(t_n)$, ale jelikož numerické metody vedou ke konstrukci pouze řešení přibližného zvolili jsme označení nalezených numerických hodnot řešení písmenem v, čímž je odlišíme od přesných hodnot u. Podobně zavedeme značení $f_n = f(v_n, t_n)$ (tentokrát nebudeme potřebovat přesnou hodnotu $f(u(t_n), t_n)$).

Příklad 1 (Eulerova metoda): Asi první myšlenka co člověka napadne pro numerické nalezení řešení rovnice (4.1) je nahradit derivaci konečnou diferencí

$$\frac{u_{n+1} - u_n}{h} \simeq u'(t_n) = f_n.$$

V rovnici je psána jen přibližná rovnost, neboť jak jsme si ukázali výše je chyba nahrazení derivace zlomkem na levé straně rovna $O(h)$. Přibližnou rovnost nahradíme přesnou a výsledek interpretujeme jako předpis jak postupně konstruovat posloupnost hodnot $v_1, v_2, \ldots$ z počáteční hodnoty $v_0 = u_0$ pomocí vztahu (EU)

$$v_{n+1} = v_n + hf_n.$$

Příklad 2 (Implicitní/zpětná Eulerova metoda): Toto metodu nalezneme drobnou modifikací předchozího postupu. Využijeme toho, že první diferencí můžeme aproximovat hodnotu derivace v obou krajních bodech se stejnou chybou (tj. O(h)) a tedy

$$\frac{u_{n+1} - u_n}{h} \simeq u'(t_{n+1}) = f_{n+1}.$$

Pro konstrukci přibližného řešení dostaneme předpis (IE)

$$v_{n+1} = v_n + hf_{n+1}.$$

Zde hovoříme o implicitní metodě, neboť v_{n+1} nalezneme z v_n až po vyřešení rovnice

$$v_{n+1} - hf(v_{n+1}, t_n) = v_n.$$

Pro konkrétní pravou stranu $f(u,t)$ může jít tato rovnice vyřešit analyticky, ale obecně jsme odkázáni na iterační metody. O konkrétní implementaci těchto iterací se ještě podrobněji zmíníme, zatím se spokojme s předpokladem, že tuto rovnici umíme pro novou hodnotu v_{n+1} vyřešit přesně.

Příklad 3 (metody zpětné diference vyššího řádu): Odvození implicitní Eulerovy metody (IEU) výše vychází vlastně z aproximace operátoru derivace na pravidelné síti do prvního řádu, tak jak jsme o něm mluvili v kapitole ... tj.

$$hDu_{n+1} = -\ln(I - \nabla)u_{n+1} \simeq \nabla u_{n+1} = u_{n+1} - u_n.$$

rozvojem logaritmu to Taylorova rozvoje vyššího řádu dostaneme přesnější schemata. Ukážeme si to na příkladu metody zpětné diference 3.řádu. Nejdříve napíšeme

$$hf_{n+1} = hDu_{n+1} = -\ln(I - \nabla)u_{n+1} \simeq (\nabla + \tfrac{1}{2}\nabla^2 + \tfrac{1}{3}\nabla^3)u_{n+1}.$$

Odtud explicitním vyjádřením mocnin operátoru zpětné diference dostaneme schema (ZD3)

$$v_{n+1} = \tfrac{18}{11}v_n - \tfrac{9}{11}v_{n-1} + \tfrac{2}{11}v_{n-2} + \tfrac{6}{11}hf_{n+1}$$

s lokální chybou $O(h^4)$. Předchozí příklady ukazovali tzv. jednokrokové metody, které vyjadřují hodnotu v_{n+1} pomocí jediné předchozí hodnoty v_n. Tato metoda je příkladem (implicitní) tříkrokové metody. Pro konstrukci v_{n+1} potřebujeme znát tři předchozí hodnoty v_n, v_{n-1} a v_{n-2}. To může představovat obtíž pro nastartování rekurentní relace pro nějž potřebujeme tři hodnoty v_0, v_1, v_2. Počáteční podmínka ovšem fixuje jen v_0. Pokud zadání úlohy neumožňuje nějaký speciální trik (např. Taylorův rozvoj řešení v okolí počátku) můžeme hodnoty v_1 a v_2 dostat použitím jednokrokové metody, ale tím pokazíme lokální diskretizační chybu. Řešením je použitím nelineárních jednokrokových metod typu Runge-Kuta, o nichž si povíme později.

Příklad 4 (Lichoběžníkové pravidlo/ metoda Crank-Nicolsonova): Poslední dva důležité příklady lze odvodit kvadraturními formulemi. Můžeme totiž psát

$$u_{n+1} - u_n = \int_{t_n}^{t_{n+1}} f(u(t), t)dt \equiv Jf_n.$$

Různými aproximacemi integrálu Jf_n pak dostáváme různá diferenční schemata. Například použitím lichoběžníkového pravidla $Jf_n = h(f_n + f_{n+1})/2$ dostáváme metodu Cranka a Nicolsona (CN)

$$v_{n+1} = v_n + \tfrac{1}{2}hf_n + \tfrac{1}{2}hf_{n+1}.$$

Jde o implicitní jednokrokovou metodu jako (EU), ale s přesností druhého řádu (lokální diskretizační chyba o řád větší $O(h^3)$). Použitím aproximací operátoru J rozvojem do zpětných diferencí vyššího řádu lze dostat vícekrokové metody vyššího řádu přesnosti (Adamsovy metody). Později si ukážeme ještě elegantnější metodu jak tyto metody odvodit pomocí aproximace funkce logaritmus podílem polynomů.

4.2 Lineární mnohakrokové metody

Uvedli jsme několik metod jak konstruovat přibližná řešení počáteční úlohy pro zadanou diferenciální rovnici. Všechny tyto metody spadají do třídy metod, které budeme souhrně nazývat lineární multikrokové formule (LIMUFO).

Definice (*LIMUFO*): Obecná $s-$kroková lineární multikroková formule (metoda) je zadána vozrcem

$$\sum_{j=0}^{s} \alpha_j v_{n+j} = h \sum_{j=0}^{s} \beta_j f_{n+j}, \tag{4.2}$$

kde $\alpha_s = 1$ a $\alpha_0 \beta_0 \neq 0$. Pokud navíc $\beta_s = 0$ nazýváme formuli *explicitní*, jinak jde o formuli *implicitní*.

4.2.1 Přesnost a konzistence metody

LIMUFO musí integrovat libovolnou rozumnou diferenciální rovnici. To je možné jedině tak, že formuli splňuje dostatečně přesně (ve smyslu rozvoje pro $h \to 0$) libovolná funkce $u(t)$ jejíž hodnoty dosazujeme za v_n a její derivace, kterou dosazujeme za f_n. Přesnost formule lze ověřovat dosazením Taylorových rozvojů

$$\begin{aligned} u_{n+j} &= u_n + jhu' + \tfrac{1}{2}(jh)^2 u'' + ..., \\ hf_{n+j} = hu'_{n+j} &= hu' + jh^2 u'' + \tfrac{1}{2} j^2 h^3 u''' + ... \end{aligned}$$

do lineární multikrokové formule. Všiměte si, že vždy derivace řádu k, se vyskytuje vynásobená mocninou h^k a tedy

$$\mathcal{L}u_n \equiv \sum_{j=0}^{s} \alpha_j u_{n+j} - h \sum_{j=0}^{s} \beta_j u'_{n+j} = C_0 u_n + C_1 h u'_n + C_2 h^2 u''_n + ...,$$

kde jsme současně zavedli lineární multikrokový operátor $\mathcal{L}$ a koeficienty rozvoje C_i. Lineární multikroková formule bude integrovat diferenciální rovnice tím lépe, čím bude pravá strana menší, tj. čím více koeficientů C_i se nám podaří vynulovat.

Definice (*konzistence a řád přesnosti LIMUFO*): Nechť koeficienty v rozvoji lineárního multikrokového operátoru splňují $C_0 = C_1 = ... = C_p = 0$ a $C_{p+1} \neq 0$. Pak řekneme, že LIMUFO je *řádu přesnosti* p a C_{p+1} je tzv. *chybová konstanta*. Řekneme že formule je *konzistentní*, pokud $C_0 = C_1 = 0$, tj. pokud řád přesnosti je alespoň 1.

Pokuste se explicitně aplikovat výše uvedené definice na některou z metod uvedených v příkladech výše. Například v Eulerově metodě jsme použili vyjádření derivace s lokální diskretizační chybou $O(h)$. Lokální chyba příslušné LIMUFO je $O(h^2)$ (rovnici jsme vynásobili h). Metoda je tudíž řádu přesnosti 1, tj. je konzistentní. Zkuste najít chybovou konstantu C_2 dosazením zmíněných Taylorových rozvojů do definice lineárního multikrokového operátoru $\mathcal{L}$. Obecně najdeme vztahy

$$\begin{aligned} C_0 &= \alpha_0 + \alpha_1 + ... + \alpha_s, && (4.3) \\ C_1 &= (\alpha_1 + 2\alpha_2 + ... + s\alpha_s) - (\beta_0 + \beta_1 + ... + \beta_s), && (4.4) \\ &\vdots && (4.5) \\ C_m &= \sum_{j=0}^{s} \frac{j^m}{m!} \alpha_j + \sum_{j=0}^{s} \frac{j^{m-1}}{(m-1)!} \beta_j, && (4.6) \end{aligned}$$

které lze použít k nalezení řádu přesnosti zadané metody, ale také ke konstrukci LIMUFO metodou neurčitých koeficientů (zvolíme si, které α a β budou nenulové a ty najdeme řešením soustavy lineárních rovnic $C_i = 0$). Ukážeme si ale elegantnější metodu jak tyto dva úkoly provést.

4.2.2 Konstrukce metod

Nejdříve si uvědomíme, že operátor $\mathcal{L}$ můžeme napsat pomocí operátoru translace a operátoru derivace

$$\mathcal{L} = \rho(E) - hD\sigma(E) = \rho(e^{hD}) - hD\sigma(e^{hD}), \tag{4.7}$$

kde jsme zavedli *charakteristické polynomy* LIMUFO $\rho(z) \equiv \sum \alpha_j z^j$ a $\sigma(z) \equiv \sum \beta_j z^j$. Definici koeficientů C_i můžeme tedy vyjádřit jako

$$\mathcal{L}(\kappa) = \rho(e^{\kappa}) - \kappa\sigma(e^{\kappa}) = C_0 + C_1\kappa + C_2\kappa^2 + ...$$

Koeficienty C_i a tedy řád dané metody můžeme tedy rychle najít pomocí Taylorova rozvoje této funkce.

Příklad 5 (řád přesnosti metod z příkladů 1-4): Pomocí programu *Maple* pro symbolické manipulace postupně zjistíme lokální diskretizační chyby metod (EU), (IE), (ZD3) a (CN):

```
>taylor(exp(x)-1-x*(1),x,3);
```

$$\tfrac{1}{2}x^2 + O(x^3),$$

```
>taylor(exp(x)-1-x*(exp(x)),x,3);
```

$$-\tfrac{1}{2}x^2 + O(x^3),$$

```
>taylor(exp(3*x)-18/11*exp(2*x)+9/11*exp(x)-2/11-x*6/11*exp(3*x),x,5);
```

$$-\tfrac{3}{22}x^4 + O(x^5),$$

```
>taylor(exp(x)-1-x/2*(exp(x)+1),x,4);
```

$$-\tfrac{1}{12}x^3 + O(x^4).$$

Ověřili jsme tedy, že metody Eulerova explicitní i implicitní metody jsou prvního řádu (tj. lokální diskretizační chyba druhého řádu), metoda zpětné diference je třetího řádu jak jsme předpokládali a metoda Crankova-Nicolsonova je druhého řádu přesnosti.

Uvedený postup lze obrátit a použít ke konstrukci multikrokových metod. Nejdříve pomocí chybového operátoru (4.7), který položíme roven nule vyjádříme operátor derivace

$$hD = \ln(E) = \frac{\rho(E)}{\sigma(E)}.$$

Vidíme tedy, že multikroková metoda úzce souvisí s aproximací funkce logaritmu podílem dvou polynomů. Tento typ aproximace se nazývá Padého aproximace (viz dodatek). Padého aproximace se provádí porovnáním Taylorova rozvoje výrazu na obou stranách. V našem případě je potřeba dostadit $E = I - \nabla$ a tedy provádět Taylorův rozvoj v bodě $E = I$.

Příklad 5 (Odvozování různých LIMUFO pomocí Padého aproximace): Z definice polynomů $\rho(z)$ a $\sigma(z)$ je rovnou vidět, že Eulerova a implicitní Eulerova metoda odpovídají aproximacím

$$\ln(E) = \ln(I - \nabla) = -\nabla - \tfrac{1}{2}\nabla^2 - \tfrac{1}{3}\nabla^3 + \cdots = \frac{\rho(I-\nabla)}{\sigma(I-\nabla)} = \frac{\alpha_0 + \alpha_1 E + ... + \alpha_s E^s}{\beta_0 + \beta_1 E + ... + \beta_s E^s}$$

Shrnutí nejpoužívanějších metod je uvedeno v tabulce

```
>simplify(pade(ln(z), z = 1, [1, 1]));
```

$$\frac{2(z-1)}{z+1}$$

```
>simplify(pade(ln(z)/z^3, z = 1, [1, 3]));
```

$$\frac{24(z-1)}{-9 + 37z - 59z^2 + 55z^3}$$

```
>simplify(pade(ln(z)/z^3, z = 1, [1, 4]));
```

$$\frac{720(z-1)}{-19 + 106z - 264z^2 + 646z^3 + 251z^4}$$

Tabulka I: Explicitní metody typu Adams-Bashforth

(EU)	$v_{n+1} = v_n + hf_n$
(AB2)	$v_{n+1} = v_n + h(\frac{3}{2}f_n - \frac{1}{2}f_{n-1})$
(AB3)	$v_{n+1} = v_n + h(\frac{23}{12}f_n - \frac{16}{12}f_{n-1} + \frac{5}{12}f_{n-2})$
(AB4)	$v_{n+1} = v_n + h(\frac{55}{24}f_n - \frac{59}{24}f_{n-1} + \frac{37}{24}f_{n-2} - \frac{9}{24}f_{n-3})$

Tabulka II: Implicitní metody typu Adams-Moulton

(IE)	$v_{n+1} = v_n + hf_{n+1}$
(CN)	$v_{n+1} = v_n + h(\frac{1}{2}f_{n+1} + \frac{1}{2}f_n)$
(AM3)	$v_{n+1} = v_n + h(\frac{5}{12}f_{n+1} + \frac{8}{12}f_n - \frac{1}{12}f_{n-1})$
(AM4)	$v_{n+1} = v_n + h(\frac{9}{24}f_{n+1} + \frac{19}{24}f_n - \frac{5}{24}f_{n-1} + \frac{1}{24}f_{n-2})$
(AM5)	$v_{n+1} = v_n + h(\frac{251}{720}f_{n+1} + \frac{646}{720}f_n - \frac{264}{720}f_{n-1} + \frac{106}{720}f_{n-2} - \frac{19}{720}f_{n-3})$

Tabulka III: Explicitní metody založené na zpětné diferenci

(IE)	$v_{n+1} = v_n + hf_{n+1}$
(BD2)	$v_{n+1} = \frac{4}{3}v_n - \frac{1}{3}v_{n-1} + \frac{2}{3}hf_{n+1}$
(BD3)	$v_{n+1} = \frac{18}{11}v_n - \frac{9}{11}v_{n-1} + \frac{2}{11}v_{n-2} + \frac{6}{11}hf_{n+1}$
(BD4)	$v_{n+1} = \frac{48}{25}v_n - \frac{36}{25}v_{n-1} + \frac{16}{25}v_{n-2} - \frac{3}{25}v_{n-3} + \frac{12}{25}hf_{n+1}$

4.2.3 Konvergence a stabilita

V minulé sekci jsme si ukázali, že nejpřesnější dvoukroková explicitní formule je

$$v_{n+1} = -4v_n + 5v_{n-1} + h(4f_n + 2f_{n-1}), \qquad (*) \tag{4.8}$$

která je třetího řádu přesnosti. Na obrázku ??a je ukázáno řešení rovnice $u'(t) = u$ s počáteční podmínkou $u(0) = 1$ na intervalu $t \in \langle 0, 1\rangle$ jednak pomocí této metody a také pomocí metody Adamse-Bashfortha druhého řádu s krokem $h = 0.1$. Je vidět, že metoda (*), ač formálně vyššího řádu přesnosti, nedává dobré výsledky. Obrázek ??b ukazuje, že ani zjemňování kroku h nevede pro metodu (*) k lepším výsledkům. Naopak, zatímco chyby Bashforthových method (AB2) a (AB4) se chovají jako $O(h^2)$ a $O(h^4)$, jak mají podle tvaru formální diskretizační chyby, metoda (*) má se zmenšujícím se krokem stále větší chybu.

Evidentně ne každá konzistentní multikroková metoda konverguje. Příčinu je třeba hledat ve stabilitě metody. Uvedený příklad je volen tak, že výsledný rekurentní vztah (*) můžeme pro $f(u) = u$ vyřešit pomocí metod na řešení diferenčních rovnic z kapitoly 3. V našem případě tedy (*) dává relaci

$$v_{n+2} + 4v_{n+1} - 5v_n = 4hv_{n+1} + 2hv_n,$$

jehož obecným dvě lineárně nezávislá řešení jsou

$$v_n^{(1/2)} = \lambda_{1/2}^n,$$

kde λ_1 a λ_2 jsou kořeny charakteristického polynomu

$$\lambda^2 + 4(1-h)\lambda - 5 + 2h = 0.$$

Pro malé hodnoty h najdeme kořeny $\lambda_1 = 1 + h + O(h^2)$ a $\lambda_2 = -5 + 3h + O(h^2)$. Snadno nahlédneme, že první řešení diferenční rovnice odpovídá správnému řešení diferenciální rovnice $u(t) = \exp(t)$, neboť pro zmenšující se délku kroku $h = 1/N$ dostaneme

$$v_N = \lambda_1^N = (1 + \frac{1}{N})^N \xrightarrow{N\to\infty} e = u(1).$$

Důvodem toho, že metoda nekonverguje je existence druhého řešení, které se pro malá h chová jako $v_n = \lambda_2^2 \simeq (-5)^n$. Tomuto řešení budeme říkat parazitní řešení. I když volbou počáteční podmínky vybereme správné řešení, zaokrouhlovacími chybami vždy přibereme malou část parazitního řešení. Toto řešení exponenciálně roste a brzy přeroste správné řešení. Tento problém se se zmenšováním kroku zhoršuje. Zvolená metoda proto nekonverguje. Podmínkou konvergence, je že všechna parazitní řešení jsou omezená. Ještě si uvědomme, že rekurentní relace výše je v limitě $h \to 0$ daná jen koeficienty α_s, tj. polynomem $\rho(z)$. Z toho vychází následující definice a věta.

Definice (*stabilita LIMUFO*): Lineární multikrokovou formuli nazýváme *stabilní* pokud jsou všechna řešení rekurentní relace $\rho(E)v_n = 0$ omezená pro $n \to \infty$.

Z teorie řešení lineárních diferenčních rovnic z kapitoly 3, plyne ihned následující věta.

Věta: Lineární multikroková formule je stabilní právě když všechny kořeny z_i polynomu $\rho(z)$ splňují podmínku $|z_i| \leq 1$ a případné kořeny pro něž $|z| = 1$ jsou jednoduché.

Je dobré si uvědomit následující:

- Charakteristický polynom $\rho(z)$ má vždy kořen $z = 1$. Platí totiž $\rho(1) = C_0$ (viz 4.3) tedy existence tohoto kořenu plyne z konzistence metody. Tomuto kořenu říkáme principiální a to je kořen, který vede na řešení rovnice. Ostatní kořeny jsou parazitní.

- Adamsovy metody představené v předchozím odstavci jsou stabilní, neboť $\rho(z) = z^s - z^{s-1} = z^{s-1}(z-1)$. Kromě principiálního kořenu mají tudíž jen vícenásobný kořen $z = 0$.

- Podrobnějším zkoumáním s-krokové metody vycházející ze zpětné diference (rovněž studované v předchozím odstavci) zjistíme, že metoda je stabilní pro $1 \leq s \leq 6$ a nestabilní pro $z \geq 7$.

Obecně nutné podmínky stability LIMUFO shrnuje následující věta

Věta (*PRVNÍ DAHLQUISTOVA BARIÉRA STABILITY*): Každá stabilní s-kroková lineární multikroková formule řádu přesnosti p plňuje

$$p \leq \begin{cases} s+2 & \dots s \text{ sudé} \\ s+1 & \dots s \text{ liché} \\ s & \dots \text{pro explicitní formuli} \end{cases}$$

V příkladě výše jsme viděli, že LIMUFO (*) nekonverguje, protože není stabilní. Otázka je, jestli každá stabilní metoda (dost vysokého řádu přesnosti) už konverguje. Ukazuje se, že ano. To a rychlost konvergence řeší následující věty, které uvedeme opět bez důkazu.

Definice (*konvergence LIMUFO*): Řekneme, že lineární multikroková metoda je konvergentní, když pro všechny rozumné (viz [3]) počáteční problémy a pro startovací hodnoty splňující $\|v_n - u_0\| \xrightarrow{h\to 0} 0$ pro všechna $n = 0, ..., s-1$ platí $\|v(t) - u(t)\| \xrightarrow{h\to 0} 0$.

Věta (*DAHLQUISTOVA VĚTA O EKVIVALENCI*): Lineární multikroková metoda je konvergentní právě když je konzistentní a stabilní.

Tato věta tedy řeší otázku konvergence LIMUFO převedením na vyšetřování stability, tj. na hledání kořenů charakteristického polynomu $\rho(z)$ a konzistence. Konzistenci jsme definovali tak, že řád přesnosti konvergentní metody je alepoň $p = 1$. Lokální diskretizační chyba tedy je alespoň $O(h^2)$, tj. po provedení $N \sim 1/h$ kroků se nasčítá na $O(h) \to 0$ (pokud je metoda stabilní). Dahlquistova věta o ekvivalenci tedy říká, že tento naivní odhad je správný. Otázka je, jestli podobný odhad platí i pro metodu vyšší přesnosti. Tam bychom naivně očekávali, že lokální chyba $O(h^{p+1})$ se nasčítá na globální chybu $O(h^p)$. Dá se ukázat, že pro stabilní metody je to pravda.

Věta (*o globální přesnosti*): Nechť je zadána ROZUMNÁ počáteční úloha a $f(u,t)$ je navíc spojitě diferencovatelná, a nechť posloupnost v_n je spočtena konvergentní LIMUFO řádu přesnosti alespoň p a nechť startovací hodnoty pro LIMUFO jsou zvoleny tak, že splňují $\|v_n - u(t_n)\| = O(h^p)$ pro $h \to 0$ a $0 \leq n \leq s-1$. Pak $\|v(t) - u(t)\| = O(h^p)$ pro $\to 0$ stejnoměrně na $t \in \langle 0, \tau \rangle$.

4.2.4 Úlohy se silným tlumením a oblast stability

V minulém odstavci jsme vyjasnili podmínky za jakých je daná LIMUFO konvergentní, tj. kdy dostaneme v limitě $h \to 0$ správné řešení zadané počáteční úlohy. V praxi ovšem potřebujeme vědět nejen, že limita pro $h \to 0$ je správná, ale že pro konečné hodnoty h dostáváme rozumné výsledky. To může být problematické zvláště pro tzv. úlohy se silným tlumením, kde se ukazuje, že stabilní LIMUFO dává nestabilní výsledky pokud není h opravdu extrémně malé. Ukážeme si to na příkladu (Trefethen [3]).

Příklad: Řešte numericky rovnici

$$u'(t) = -100[u - \cos(t)] - \sin(t)$$

s počáteční podmínkou $u(0) = 1$. Řešením úlohy očividně je $u(t) = cos(t)$. Podívejme se na to jak se s úlohou vypořádají dvě metody druhého řádu: Adamsova-Bashforthova (AB2) a zpětná diference (BD2). Obě metody jsou dvoukrokové a tedy kromě počáteční podmínky $v_0 = 0$ protřebujeme ještě v_1, kam si pro jednoduchost dosadíme přesné řešení $v_1 = cos(h)$. Obrázek ?? ukazuje výslednou chybu řešení $\Delta u = abs(v(1) - u(1))$ v bodě $t = 1$ pro přibližné řešení získané metodami AB2 a BD2 pro různé hodnoty délky kroku h. Jak bychom očekávali z předchozí kapitoly, chyba řešení jde k nule pro malé hodnoty kroku h pro obě metody, přičemž asymptoticky se chyba chová jako $O(h^2)$. Rozdíl je v tom jak rychle se každá z metod k této asymptotice dostane. Zatímco metoda používající zpětné diference dává rozumné výsledky s poměrně velkým krokem, formule Adamse a Bashfortha nefunguje dobře pokud $h > 0.01$.

Příklad: Podobné problémy pro soustavu rovnic

$$\begin{aligned} u' &= -5u + 6v, \\ v' &= 4u - 5v. \end{aligned}$$

Rozbor a obrázek doplním později, ale opět jde od dvě rozdílné časové škály, dané vlastními čísly matice pravé strany $\lambda = -0.1, -9.9$

Důvod popsaného chování pochopíme, když zopakujeme analýzu stability LIMUFO tentokrát při konečné hodnotě h. Analýza při nulovém h nezávisela na pravé straně $f(u,t)$. Tentokrát se nevyhneme charakteristice funkce f. Obvyklým trikem je linearizace funkce $f(u,t)$ v okolí bodu (u,t), kde nás zajímá stabilita: $f(u,t) \approx au + b$, přičemž konstantní člen b můžeme ignorovat, neboť při vyšetřování stability příslušných diferenčních rovnic přispívá pouze k partikulárnímu a ne obecnému řešení. Dosazením této aproximace do LIMUFO dostaneme rekurentní relaci

$$\sum_{j=0}^{s} (\alpha_j - \tilde{h}\beta_j) v_{n+j} = 0, \tag{4.9}$$

kde jsme zavedli značení $\tilde{h} = ah$

Definice: (*absolutní stabilita LIMUFO*)
Řekneme, že lineární multikroková formule je absolutně stabilní pro dané $\tilde{h} \in \mathbb{C}$ pokud každé řešení rekurentní relace (4.9) je omezené pro $n \to \infty$.

Z teorie řešení lineárních diferenčních rovnic přitom dostaneme následující charakteristiku absolutní stability:

Věta: (*Polynom stability*)
Lineární multikroková formule je absolutně stabilní, právě když všechny kořeny z **polynomu stability**

$$\pi_{\tilde{h}}(z) \equiv \sum_{j=0}^{s} (\alpha_j - \tilde{h}\beta_j) z^j = \rho(z) - \tilde{h}\sigma(z) \tag{4.10}$$

splňují podmínku $|z| \leq 1$ a kořeny s $|z| = 1$ jsou jednoduché.

Definice: (*oblast stability*) Oblast stability S lineární multikrokové metody definujeme, jako množinu všech bodů $\tilde{h} \in \mathbb{C}$, pro něž je daná formule absolutně stabilní.

Oblast stability pak dává návod jak volit velikost kroku h pro konkrétní problém a metodu. Je prostě potřeba volit tak malé h, aby příslušné $\tilde{h} = ah$ bylo v oblasti stability. Poznamenejme, že samotná stabilita metody, kterou jsme vyšetřovali v předchozím odstavci zaručuje, že bod $\tilde{h} = 0$ patří do oblasti stability. Oblasti stability pro metody Adamsovy a zpětné diference jsou vyznačeny na obrázcích ??-??(ukazoval jsem na přednášce, časem doplním). Z obrázků je

zřejmý význam implicitních metod, které mají větší oblast stability a především metody zpětné diference, jejíž oblast stability dokonce obsahuje celou zápornou poloosu.

Pojem absolutní stability není vhodný pro vyšetřování rostoucích řešení, tj. $a > 0$ (dají se zavést jiné pojmy, např. L-stabilita), neboť pak samotné principiální řešení není omezené. Při vyšetřování pojmu stability formule (který by neměl záviset na volbě f) se proto omezíme na $\tilde{h}$ z levé poloroviny v komplexní rovině.

Definice: (*A-stabilita*)
Řekneme, že daná metoda je A-stabilní, pokud celá levá komplexní polorovina $\mathrm{Re}\tilde{h} < 0$ patří do oblasti stability. Řekneme, že metoda je A(α) stabilní, pokud množina $|\mathrm{Arg}\,\tilde{h}| > \pi - \alpha$ (výseč kolem záporné reálné osy s vrcholovým úhlem 2α) patří do oblasti stability.

Ukazuje se, že A-stabilita je velmi tvrdé kritérium. Přitom pro úlohy se silným tlumením, zvláště pro velké soustavy diferenciálních rovnic je důležitá. Platí následující věta.

Věta: (*DRUHÁ DAHLQUISTOVA BARIÉRA STABILITY*)
Řád p A-stabilní lineární multikroková formule musí splňovat nerovnost $p \leq 2$. Explicitní formule nemůže být A-stabilní.

Poznámka: Formule založené na zpětné diferenci jsou A(0) stabilní pro $p \leq 6$ (přesněji A(α) stabilní přičemž $\alpha = \pi$ pro $s = 1, 2$ a $\alpha \simeq 86°, 73°, 52°$ a $18°$ pro $s = 3, 4, 5$ a 6)

Poznámka: Z druhé Dahlquistovy bariéry stability plyne význam metod prvního a druhého řádu pro řešení parciálních diferenciálních rovnic, které se často prostorovou diskretizací převedou na obrovskou soustavu obyčejných diferenciálních rovnic v časové oblasti.

4.3 Nelineární jednokrokové metody typu Runge-Kutta

Zatím jsem nestihl přepsat poznámky do LaTeXu. Doporučuji se podívat do numerických receptů [1] nebo opět do Trefethenových poznámek [3]. Metody typu Runge-Kutta mají tu výhodu, že jsou jednokrokové a k jejich nastartování tedy stačí pouze počáteční podmínka. Vyššího řádu přesnosti dosahují tím, že si před provedení kroku z v_n do v_{n+1} "osahají" funkci $f(u,t)$ v několika bodech v okolí bodu $(u,t) = (v_n, t_n)$ (metody vyššího řádu samozřejmě osahávají více bodů, a tudíž na provedení jednoho kroku potřebují více vyčíslení funkce f). Výsledná formule pro provedení jednoho kroku je nelineární (pro nelineární funkci f). Opět lze vyšetřovat oblast stability Runge-Kuttových metod. Ukazuje se, že nejsou A-stabilní, ale oblast stability vždy obsahuje alespoň kruh $|z + 1| \leq 1$ v komplexní rovině.

4.4 Numerovova metoda - řešení rovnic druhého řádu

Tato kapitola se zabývá jistou speciální metodou na rovnice tvaru

$$u''(t) = f(u,t).$$

Zatím je ponechána dle staré verze poznámek. Musím ji přepsat v duchu teorie LIMUFO. Zájemcům to zatím ponechávám jako cvičení. Jen drobná nápověda: Lokální diskretizační chyba bude nyní souviset s operátorem

$$\mathcal{L} = \rho(E) - (hD)^2\sigma(E)$$

. Numerovovu metodu pak lze chápat jako aproximaci funkce $(\ln z)^2$ podílem polynomů

$$12\frac{z^2 - 2z + 1}{z^2 + 10z + 1}$$

což dává lokální chybu 6. řádu. Stabilitu je třeba definovat trochu mírněji, neboť pro rovnici druhého řádu musí nutně existovat dvě principiální řešení, které v limitě $h \to 0$ vedou na dva kořeny charakteristického polynomu s $z = 1$. S tím souvisí také to, že lokální diskretizační chyba 6. řádu vede nakonec na globální chybu pouze 4. řádu a ne 5. jak by člověk čekal na základě předchozí kapitoly.

A ZDE JSOU SLÍBENÉ STARÉ POZNÁMKY:

Numerovova metoda je speciální numerickou metodou pro diskretizaci rovnice druhého řádu tvaru

$$y''(x) + k^2(x)y(x) = S(x), \tag{4.11}$$

kde $y(x)$ je neznámá funkce; $k^2(x)$ a $S(x)$ jsou předem zadané, dostatečně hladké funkce. Rovnice tohoto typu se často objevuje ve fyzice při řešení Schödingerovy nebo Poissonovy rovnice. Sestavme diskretizovanou verzi této rovnice na síti bodů x_0, $x_1, \dots$ x_N pokrývající interval $\langle a, b\rangle = \langle x_0, x_N\rangle$ rovnoměrně s krokem $h = (b-a)/N$. Nejprve napíšeme Taylorův rozvoj $y_{i\pm1}$ kolem bodu x_i

$$y_{i\pm1} = y_i \pm hy_i' + \frac{1}{2}h^2y_i'' \pm \frac{1}{3!}h^3y_i''' + \frac{1}{4!}h^4y_i^{iv} \pm \frac{1}{5!}h^5y_i^{v} + O(h^6). \tag{4.12}$$

Odtud ihned vidíme, že

$$\frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} = y_i'' + \frac{1}{12}h^2y_i^{iv} + O(h^4) \tag{4.13}$$

a tedy

$$y_i'' = \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} - \frac{1}{12}h^2y_i^{iv} + O(h^4). \tag{4.14}$$

Stejně bychom pomocí Taylorova rozvoje $y''(x)$ dostali čtvrtou dervaci

$$y_i^{iv} = \frac{y_{i+1}'' - 2y_i'' + y_{i-1}''}{h^2} + O(h^2) = \frac{(S_{i+1} - k_{i+1}^2y_{i+1}) - 2(S_i - k_i^2y_i) + (S_{i-1} - k_{i-1}^2y_{i-1})}{h^2} + O(h^2), \tag{4.15}$$

kam jsme za druhou derivaci dosadili $y'' = S - k^2y$, z rovnice (4.11). Vložením tohoto výrazu pro čtvrtou derivaci do (4.13) a drobnými úpravami dostaneme nakonec

$$\left(1 + \frac{h^2}{12}k_{n+1}^2\right)y_{n+1} - 2\left(1 - \frac{5h^2}{12}k_n^2\right)y_n + \left(1 + \frac{h^2}{12}k_{n-1}^2\right)y_{n-1} = \frac{h^2}{12}(S_{n+1} + 10S_n + S_{n-1}) + O(h^6). \tag{4.16}$$

Tuto formuli lze chápat jako vyjádření y_{n+1} pomocí y_n a y_{n-1} a umožňuje generovat postupně celé řešení rovnice (4.11) z hodnot y_0, y_1. Čísla y_0, y_1 musíme určit z počáteční podmínky, t. j. ze zadaných hodnot $y(x)$ a $y'(x)$ v bodě $x = x_0$.

Lokální diskretizační chyba Numerovovy metody je $O(h^6)$. Naivní úvahou bychom očekávali, že globální chyba po N-násobném opakování použití lokálního vzorce bude $NO(h^6) = O(h^5)$, neboť $N = (x_N - x_0)/h$. Podrobnější analýza ukazuje, že chyba ve skutečnosti naroste ještě o řád více na $O(h^4)$ a řešení získané Numerovovou metodou je správné do řádu h^3.

Pokud chceme určit čísla y_0, y_1 z počáteční podmínky y_0, y_0' a nepokazit přitom řád diskretizační chyby, můžeme postupovat podobně jako výše, ale místo druhé derivace si rozdílem Taylorových rozvojů v bodech x_{i+1} a x_{i-1} vyjádříme druhou derivaci a do vzorce dosadíme výraz pro třetí derivaci získaný derivováním (4.11). Pro $i = 0$ dospějeme ke vztahu

$$2hy'(x_0) - \frac{h^2}{6}(S_1 - S_{-1}) = \left(1 + \frac{h^2}{6}k_1^2\right)y_1 - \left(1 + \frac{h^2}{6}k_{-1}^2\right)y_{-1}. \tag{4.17}$$

Tato rovnice tvoří spolu s diferenčním schematem (4.16) soustavu dvou rovnic pro dvě neznámé y_1 a y_{-1}. Jejím řešení dostáváme vzorec

$$y_1 = \frac{\left(1+\frac{h^2}{12}k_{-1}^2\right)\tilde{y_0} + \left(1+\frac{h^2}{6}k_{-1}^2\right)\tilde{y_1}}{\left(1+\frac{h^2}{12}k_1^2\right)\left(1+\frac{h^2}{6}k_{-1}^2\right) + \left(1+\frac{h^2}{12}k_{-1}^2\right)\left(1+\frac{h^2}{6}k_1^2\right)}, \tag{4.18}$$

$$\tilde{y_0} = \left(2-\frac{5h^2}{6}k_0^2\right)y_0 + \frac{h^2}{12}(S_1+10S_0+S_{-1}), \tag{4.19}$$

$$\tilde{y_1} = 2hy_0' + \frac{h^2}{6}(S_1-S_{-1}). \tag{4.20}$$

Použití těchto těžkopádných vzorců se lze často vyhnout. Pokud zní jedna z počátečních podmínek $y(x_0)=0$, potom druhá podmínka $y'(x_0)=y_0'$ určuje celkovou normalizaci funkce $y(x)$. Pokud nás tato normalizace nezajímá, nebo pokud ji budeme určovat až nakonec z nalezených hodnot funkce, můžeme jako druhou okrajovou podmínku vzít $y_1=1$, čímž se vyhneme použití předchozího vzorce.

Kapitola 5

Numerická lineární algebra

TO JSEM JEŠTĚ NENAPSAL. PŘEDNÁŠKU JSEM PŘIPRAVOVAL DLE UČEBNICE [2] KTEROU SI MŮŽETE STÁHNOUT NA ADRESE (ČÍSLO 05 JE AKTUÁLNÍ MĚSÍC - ADRESA SE MĚNÍ)

`http://utf.mff.cuni.cz/~houfek/esources05/Books/Numerical Methods, Programming, Software/`

KDE JDE ZVLÁŠTĚ O KAPITOLY I.3-5, II.6-8,10,11, III.(BEZ DŮKAZŮ), IV., V.24-26 DÁLE DIAGONALIZACE JACOBIHO METODOU PODLE NUMERICKÝCH RECEPTŮ

5.1 Úvod. Vektory, matice, normy.

5.2 Faktorizace matic. Gramova-Schmidtova ortogonalizace.

5.2.1 Zpětná stabilita

5.3 Soustavy lineárních rovnic

5.4 Diagonalizace matic a úvod do iteračních metod

5.4.1 Opakování: základní fakta

Podobnostní transformace
Vlastní čísla a vektory
Charakteristický polynom, násobnost kořenů
Jordanův tvar
Schurova faktorizace

5.4.2 Givensova rotace a Jacobiho algoritmus

5.4.3

Kapitola 6

Diskrétní Fourierova trasformace a spektrální metody

OPĚT JSEM JEŠTĚ NENAPSAL. PRO TEORII DOPORUČUJI [3] KAPITOLA 2 A NUMERICKÉ RECEPTY KAPITOLA O FFT A APLIKACI NA SINOVOU - COSINOVOU TRANSFORMACI, PRO PRAKTICKOU IMPLEMENTACI

6.1 Diskrétní Fourierova transformace, vlastnosti

6.2 Algoritmus FFT

6.3 Aplikace a obecné poznámky o spektrálních metodách

[1], [4]

Příloha A

Aproximace Padé

LETOS NEZKOUŠÍM. JINAK LZE NAJÍT V NUMERICKÝCH RECEPTECH A V KNIZE KUKULINA, KRASNOPOLSKEHO A HORACKA: THEORY OF RESONANCES.

Aproximací Padé rozumíme nahrazení funkce $f(x)$ racionální funkcí

$$R_{[M,N]}(x) = \frac{P_M(x)}{Q_N(x)}, \tag{A.1}$$

kde

$$\begin{aligned} P_M(x) &= p_0 + p_1 x + p_2 x^2 + ... + p_M x^M, \\ Q_N(x) &= q_0 + q_1 x + q_2 x^2 + ... + q_N x^N \end{aligned}$$

jsou polynomy stupně M respektive N. Existuje několik variant Padého aproximace podle toho jakým způsobem určíme koeficienty p_i, q_i pro danou funkci $f(x)$. Padého aproximace pak tak může sloužit jako rozvoj kolem daného bodu x_0, interpolace danými body x_0, x_1, ..., x_K, nebo jako interpolace ve smyslu nejmenších čtverců. U těchto možností se podrobněji zastavíme níže. Nejdříve několik obecnějších poznámek.

Za první si povšimněme, že ne všechny z $M + N + 2$ koeficientů p_i, q_i jsou nezávislé. Konkrétně bez újmy na obecnosti můžeme zvolit $q_0 = 1$, neboť čitatel i jmenovat můžeme vydělit společným faktorem. Zbývá $K = M + N + 1$ koeficientů, které již jsou nezávislé.

Na rozdíl od aproximace funkce $f(x)$ polynomem je aproximant $R_{[M,N]}(x)$ nelineární funkcí koeficientů p_i, q_i. To komplikuje analýzu přesnosti a konvergence Padého aproximace a výsledky v této oblasti jsou daleko komplikovanější a neúplné. Často však ...

A.0.1 Aproximace Padé I.druhu—rozvoj v okolí bodu

Literatura

[1] Press, Teukolsky, Vetterling, Flannery: Numerical Recipes in C, Fortran...

[2] L. N. Trefethen, D. Bau III: Numerical Linear Algebra. SIAM 1997.

[3] L. N. Trefethen: Finite Difference and Spectral Methods for Ordinary and Partial Differential Equations, unpublished text, 1996, k dispozici na `http://www.comlab.ox.ac.uk/nick.trefethen/pdetext.html`

[4] L. N. Trefethen: Spectral Methods in MATLAB (SIAM 2001)

[5] Isaacson, Keller: Analysis of Numerical Methods.

[6] Koonin: Computational Physics.

[7] Henrici: Essentials of Numerical Analysis with Pocket Calculator Demonstrations.

[8] Vitásek: Numerické metody.

[9] Segethová: Základy numerické matematiky. (Skriptum MFF)

[10] Kukulin, Krasnopolski, J. Horáček, Theory of Resonances (Academia, Praha 1989).