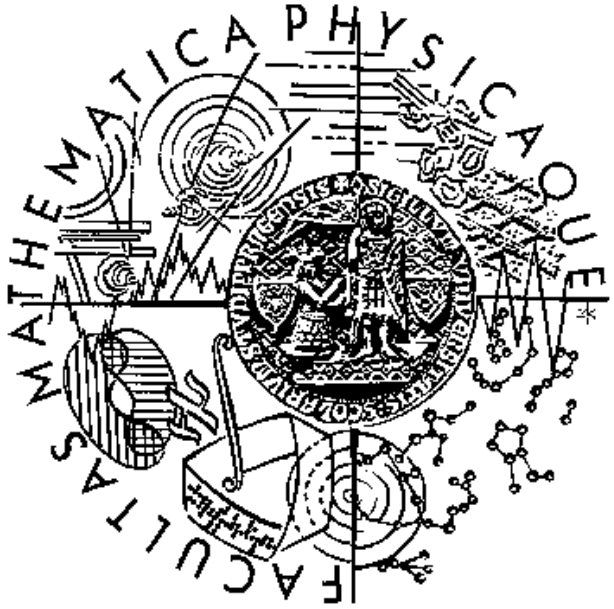
Matematicko - fyzikální fakulta
Univerzity Karlovy, Praha



Disertační práce

O některých otevřených problémech v krylovovských metodách

Petr Tichý

Školitel:	Doc. RNDr. Jan Zítko, CSc.
Konzultanti:	Doc. RNDr. Karel Najzar, CSc. Doc. Ing. Zdeněk Strakoš, DrSc.
Obor:	Vědecko-technické výpočty

WWW: http://www.cs.cas.cz/~tichy
e-mail: petr.tichy@centrum.cz
This work was supported by the Grant Agency of the Czech Republic under grant No. 201/02/0595.

Obsah

Předmluva **5**

Obsah a cíle **7**

1 Metody Krylovových prostorů **9**
1.1 Základní krylovovské metody 10
1.2 Báze Krylovových prostorů 12
1.2.1 Arnoldiho báze 14
1.2.2 Lanczosova báze 15
1.3 Metody s minimálními a galerkinovskými kvazi-rezidui 21
1.3.1 Krylovovská metoda s minimálními kvazi-rezidui 22
1.3.2 Krylovovská metoda s galerkinovskými kvazi-rezidui 25
1.3.3 Vztahy mezi metodami 26
1.4 Algoritmy krylovovských metod 26
1.4.1 Algoritmus GMRES 27
1.4.2 Algoritmus FOM 28
1.4.3 Algoritmus QMR 28
1.4.4 Algoritmus LM (BiCG) 29
1.5 Metody Lanczosova typu 31
1.5.1 Metoda CGS 33
1.5.2 Metoda BiCGStab 34
1.5.3 Metody BiCGStab2 a BiCG×MR2 35
1.5.4 Metoda BiCGStab(l) 36
1.5.5 Kvazi-minimální pojetí metody CGS (TFQMR) 36
1.6 Zastavovací kritérium 38

2 O stínovém vektoru **41**
2.1 Stínový vektor a tříkroková rekurence 42
2.2 Stínový vektor a tříkroková krylovovská metoda 45
2.3 Lanczosova metoda a rezidua jiné krylovovské metody 47
2.4 Jaký stínový vektor je optimální? 50
2.5 Metody s krátkými rekurencemi a pravděpodobnost? 52
2.6 Numerické experimenty 55

3 Odhady A-normy chyby v CG **61**
3.1 Metoda sdružených gradientů a Gaussova kvadratura 62
3.2 Základní vztahy 67
3.3 Odhad **A**-normy chyby 69

4 CG v konečné aritmetice **71**
4.1 Matematický model CG v konečné aritmetice 72
4.2 Zpoždění konvergence 74
4.3 Předpoklady a odhady zaokrouhlovacích chyb 76
4.4 Lokální ortogonalita 82
4.5 Numerické experimenty 84

5 Odhady v konečné aritmetice **90**
5.1 Odhady založené na Gaussově kvadratuře 91
5.2 Odhady založené na algebraické manipulaci 92
5.2.1 Analýza odhadu **A**-normy chyby založeného na $\nu_{j,d}$ 92
5.2.2 Analýza odhadu **A**-normy chyby založeného na $\vartheta_{j,d}$ 97
5.2.3 O odhadu **A**-normy chyby založeném na $\mu_{j,d}$ 100
5.3 Numerické experimenty 101

Dosažené výsledky, závěry a shrnutí **107**

Poděkování **109**

Seznam použité literatury **110**

Předmluva

V reálném světě vznikají problémy, které za pomocí znalostí z různých oblastí vědy dokážeme formulovat v matematickou úlohu v podobě diferenciálních a integrálních rovnic. Formulovaná úloha zpravidla nepopisuje reálný problém přesně, ale pouze přibližně. Řešení výše zmíněných rovnic leží často v nekonečně dimenzionálním prostoru a není možné jej analyticky určit. Proto hledáme aproximaci řešení v konečně dimenzionálním prostoru, provádíme diskretizaci úlohy. Po případné linearizaci před námi stojí poslední článek procesu řešení původního problému, systém lineárních rovnic

$$\mathbf{A}x = b, \tag{1}$$

jeden ze základních problémů numerické lineární algebry. Z uvedeného popisu (jeho podrobnou diskusi lze nalézt v [57]) vzniku lineárního systému (1) je zřejmé, že k efektivnímu a úspěšnému řešení původního problému je potřeba, aby jednotlivé úrovně z nichž se celý proces řešení skládá byly vzájemně propojeny a vyváženy. Jestliže je úloha na jedné úrovni aproximována s určitou přesností, nemá smysl řešit korespondující úlohu na další úrovni s výrazně odlišnou přesností. Proto je často přirozené použít k řešení systému (1) iteračních metod, které v každém kroku poskytují přibližnou aproximaci řešení a dávají nám tak možnost zastavit iterační proces na vhodné hladině přesnosti. Řídká struktura matic navíc umožňuje řešení soustav o milionech neznámých. Připomeňme, že přímé řešiče používají k nalezení řešení vždy stejného množství eliminačních kroků a v žádném z těchto kroků nemáme k dispozici přibližnou aproximaci řešení. Eliminace s sebou navíc obecně přináší zaplnění matice vedoucí, bez použití velmi složitých postupů, k obrovským nárokům na paměť počítače; výpočet se tím stává neefektivní. Uvedli jsme jeden z možných procesů vzniku systémů lineárních rovnic. Samozřejmě, že systémy lineárních rovnic mohou vznikat z různých aplikací a je nutno zvážit, které metody na jejich řešení použít. Vždy bychom ale měli mít na mysli vztah řešení lineárního algebraického systému, určeného s nějakou přesností, k původnímu problému reálného světa.

V této práci se budeme věnovat řešení systému (1) pomocí třídy iteračních metod nazývaných krylovovské metody, které se zdají být, v kombinaci s předpodmíněním, vhodné a efektivní při řešení výše popsaného problému. Intuitivně bychom na tyto metody mohli pohlížet tak, že původní mnoha dimenzionální problém projektují na podprostory malé dimenze (Krylovovy podprostory) generované pomocí opakovaných aplikací matice $\mathbf{A}$ na počáteční reziduum. V Krylovových podprostorech se rychle projevují dominantní vlastnosti matice $\mathbf{A}$ a proto bývá v příznivém případě aproximace řešení v nich hledaná dobrou aproximací i po malém počtu kroků.

Ačkoliv metody dobře fungují v praxi, nerozumíme dosud úplně jejich teoretické podstatě. K pochopení otázky proč a jak krylovovské metody fungují se snažíme dospět zkoumáním a popisem jejich konvergence v závislosti na vstupních datech úlohy. Na otázku známe odpověď v případě metody sdružených gradientů (CG), kdy je matice $\mathbf{A}$ symetrická a pozitivně definitní (viz. kapitola 3). Konvergence CG závisí na rozložení vlastních čísel matice a projekcích počátečního rezidua do jednotlivých vlastních podprostorů. V obecně nesymetrickém případě je situace komplikovanější a informace, kterou poskytují vlastní čísla, není postačující k určení konvergence, jak bylo ukázáno v [29], [27], [3]. Použijeme-li např. stabilizační techniky na jednodimenzionální konvektivně-difúzní rovnice s dominantním konvektivním členem (viz. např. [49]), je informace o rychlosti konvergence, kterou poskytují vlastní čísla operátoru výsledného diskrétního systému lineárních rovnic, zavádějící [9].

Velmi důležitou otázkou je rovněž vyhodnocování konvergence, tedy způsob, kterým se snažíme zjistit kvalitu naší aproximace, její „blízkost“ k řešení úlohy. Slovo blízkost

přitom může záviset na typu úlohy. V kapitole 3 se například zabýváme odhadem **A**-normy (energetické normy) chyby v metodě CG. V symetrickém pozitivně definitním případě má **A**-norma chyby často přímou souvislost s řešenou fyzikální či chemickou úlohou a proto je vhodná k zastavování algoritmu. V nesymetrickém případě však není obecně možné efektivně odhadnout normu chyby. Pokud z přirozenosti úlohy neplyne něco jiného, zdá se být vhodnou mírou konvergence zpětná chyba, udávající velikost maximální perturbace v matici **A** a ve vektoru pravé strany b, při níž je počítaná aproximace přesným řešením porušené soustavy [57].

Metody Krylovových prostorů můžeme rozdělit na dvě velké skupiny. Metody s krátkými rekurencemi jsou laciné (uvažujeme-li pouze cenu za iteraci) a paměťově nenáročné. Metody s dlouhými rekurencemi jsou robustnější – počítají aproximace řešení optimální v nějakém smyslu, za což musíme ovšem zaplatit většími nároky na počet operací a paměť počítače. Překvapivé je, že metody obou skupin, ačkoliv se zakládají na různých principech, dávají ideálně (v přesné aritmetice) mnohdy velmi blízké konvergenční křivky. Vztah mezi oběma skupinami metod není dosud zřejmý a jeho zkoumání by mohlo vést k samotnému pochopení konvergence metod. K této otázce se snažíme přispět v kapitole 2, ve které hovoříme o volbě stínového vektoru, jednoho z parametrů metod založených na Lanczosově procesu.

Je nutné si uvědomit, že krylovovské metody používáme téměř výhradně na počítačích s konečnou aritmetikou reprezentovanou strojovou přesností ε. Po přenesení algoritmu realizujícího danou metodu do počítače se však může stát, že hodnoty, které na základě výpočtů algoritmu v konečné aritmetice dostáváme, neodpovídají ideálním hodnotám a nesplňují teoretické vztahy dané metody. Konečná aritmetika počítače může způsobit změnu chování algoritmů a mít dokonce destruktivní účinek na konvergenci. Proto zkoumání chování algoritmů krylovovských metod v konečné aritmetice tvoří nedílnou součást jejich použití při řešení systému (1). Chování iteračních algoritmů při výpočtech s konečnou přesností je popsáno zatím jen částečně a to u algoritmů metod CG (symetrický, pozitivně definitní případ) a GMRES (obecný nesymetrický případ). V ostatních případech, zejména u algoritmů s krátkými rekurencemi, je situace zatím nejasná a numerické experimenty ukazují, že vliv zaokrouhlovacích chyb na jejich chování může být značný. Na základě souvislosti CG s Gaussovou kvadraturou vysvětlujeme v naší práci (v kapitole 4) matematický model CG v konečné aritmetice použitý v [59]. Vliv zaokrouhlovacích chyb na tuto metodu způsobí zpoždění konvergence a omezení dosažitelné přesnosti [54], [23], [31].

Další problém související s konečnou aritmetikou spočívá v získávání vhodných informací o konvergenci metody. Je důležité si uvědomit, že veškeré formule odvozené v přesné aritmetice (například konvergenční charakteristiky resp. jejich odhady) mohou v konečné aritmetice vydávat zkreslené informace o blízkosti počítané aproximace k řešení, zvláště pokud byly odvozeny na základě vlastností, které v konečné aritmetice obecně neplatí. Je tedy nutné zabývat se problémem, zda vztahy, na nichž se dané formule zakládají, platí (s nějakou malou chybou) i v konečné aritmetice. Do této problémové oblasti zasahujeme v kapitole 5, kde provádíme analýzu zaokrouhlovacích chyb ve vztazích, na nichž se zakládá odhad **A**-normy chyby v CG. Ukazujeme, kterým z odhadů můžeme věřit a kterým nikoli.

Obsah a cíle

V úvodní kapitole jsme na základě prací [50], [30] a [24] vytvořili kompaktní celek zahrnující odvození a popis nejdůležitějších krylovovských metod. Cílem této kapitoly je uvést čtenáře do problematiky krylovovských metod a ukázat, že ačkoliv existuje velmi mnoho metod a algoritmů, pracují všechny pouze na několika málo principech. Tato kapitola neobsahuje původní výsledky, snad jen osobitý náhled na danou problematiku a několik odvození provedených vlastním způsobem (viz. např. odvození QMR v oddíle 1.4.3).

Vycházíme z obecné myšlenky projektivních metod, které hledají aproximace řešení systému lineárních rovnic ve varietách definovaných počátečním přiblížením a Krylovovými podprostory. Rozdělujeme krylovovské metody na metody s minimálním a galerkinovským kvazi-reziduem používající buď Arnoldiho nebo Lanczosovu bázi, vysvětlujeme jejich výhody i nevýhody. Jedna metoda může mít různé implementace lišící se podle volby algoritmů, které realizují jednotlivé fáze výpočtu, t.j. budování báze a konstrukci aproximace řešení. Odvozujeme možné algoritmické realizace nejznámějších metod GMRES, FOM, QMR a LM. Pokračujeme metodami Lanczosova typu, které ke konstrukci aproximace řešení využívají polynomu definovaného algoritmem BiCG a dalšího pomocného polynomu. Prezentujeme algoritmy metod CGS, BiCGStab, TFQMR a vysvětlujeme principy BiCGStab2 a BiCGStab(l). V závěru diskutujeme o zastavovacím kritériu algoritmů.

Jedním z možných směrů zobecňujících metodu CG pro nesymetrické úlohy je Lanczosova metoda pro řešení nesymetrických systémů lineárních rovnic (LM), kterou se zabýváme v kapitole druhé. Cílem této kapitoly, která obsahuje původní výsledky, je přispět k řešení otevřeného problému vztahu mezi metodami s krátkými a dlouhými rekurencemi. Věříme, že klíč k zodpovězení mnoha otázek týkajících se této problematiky, je skryt v pochopení významu stínového vektoru v Lanczosově procesu a proto se věnujeme jeho zkoumání.

Ve větách 2.2, 2.3 a 2.4 rozšíříme výsledky práce [25] a určíme přesně vztah mezi krylovovskou metodou reprezentovanou tříkrokovou rekurencí a Lanczosovou metodou (LM). Vysvětlíme, že je možné určit stínový vektor tak, aby Lanczosova metoda vypočetla vybraná rezidua jiné krylovovské metody např. GMRES. Pohovoříme o tom, jaký stínový vektor lze považovat za optimální a v závěru kapitoly diskutujeme otázku proč jsou konvergenční křivky klasických krylovovských metod často velmi blízké konvergenční křivce GMRES vždy, když dochází k jejímu dostatečně rychlému poklesu. Pokusíme se vysvětlit, že náhodně volený stínový vektor je něco jiného než náhodně volené koeficiety v tříkrokové rekurenci. V numerických experimentech ověříme platnost našich teoretických výsledků a vysvětlíme, jakým způsobem volit náhodný stínový vektor. S maticí 8×8 provedeme experiment, ve kterém se pokusíme numericky určit optimální stínový vektor.

Třetí, čtvrtá a pátá kapitola zahrnuje původní výsledky zaslané k publikaci [59] (společná práce se Z. Strakošem), jejich prohloubení a rozšíření.

Ve třetí kapitole se podrobně zabýváme odhadem $\mathbf{A}$-normy chyby v metodě sdružených gradientů (CG), kde $\mathbf{A}$ je symetrická, pozitivně definitní matice. Cílem této kapitoly je vnést více světla do problému odhadování $\mathbf{A}$-normy chyby.

Na základě vztahu mezi metodou sdružených gradientů a Gaussovou kvadraturou ukážeme, jakým způsobem lze konstruovat odhady $\mathbf{A}$-normy chyby. Algebraickou cestou odvodíme nový odhad a vysvětlíme, že odhady založené na Gaussově kvadratuře jsou matematicky ekvivalentní odhadům odvozeným algebraickou cestou. Ukážeme, že nejjednodušší a nejméně početně náročný odhad je skryt ve vztazích obsažených již v původní práci [32] (numerickou stabilitou tohoto odhadu se budeme zabývat v kapitole 5).

Cílem čtvrté kapitoly je vysvětlit chování CG v konečné aritmetice a připravit podklady pro analýzu zaokrouhlovacích chyb v odhadech **A**-normy chyby.

Popíšeme základní myšlenku matematického modelu CG v konečné aritmetice založeného na pochopení CG ve smyslu Gaussovy kvadratury. Matematický model CG v konečné aritmetice nám umožní vysvětlit princip zpoždění konvergence. Formálně popíšeme zaokrouhlovací chyby vznikající při výpočtu algoritmu CG v konečné aritmetice a na základě standardního modelu aritmetiky s pohyblivou řádovou čárkou odhadneme jejich velikost. Zformulujeme a dokážeme novou větu 4.1, která se zabývá lokální ortogonalitou. V numerických experimentech se zabýváme nadhodnocením odhadů počítajících s nejhorším možným případem. Pomocí násobné aritmetiky [5] dopočteme lokální zaokrouhlovací chyby a ukážeme jejich skutečnou velikost.

V páté kapitole si klademe za cíl vysvětlit problém aplikace odhadů založených na Gaussově kvadratuře a provést analýzu zaokrouhlovacích chyb u odhadů **A**-normy chyby odvozených pomocí algebraické manipulace.

Rozšíříme analýzu zaokrouhlovacích chyb z našeho článku [59] u preferovaného odhadu a provedeme rovněž detailní analýzu zaokrouhlovacích chyb u nového odhadu. Ukážeme, které odhady lze použít i v případě, kdy jsou lokální zaokrouhlovací chyby podstatně zesilovány v průběhu výpočtu. V numerických experimentech se zabýváme analýzou platnosti vztahu, na němž se zakládá preferovaný odhad **A**-normy chyby, v konečné aritmetice. Graficky znázorňujeme chování matematicky ekvivalentních odhadů **A**-normy chyby v konečné aritmetice a vysvětlujeme praktické důsledky analýzy zaokrouhlovacích chyb. Součástí numerických experimentů bude i algebraické odvození odhadu euklidovské normy chyby a numerická demonstrace funkčnosti tohoto odhadu (analýza zaokrouhlovacích chyb pro tento odhad již přesahuje rámec naší práce).

1

Metody Krylovových prostorů

V této kapitole se zabýváme přehledem metod Krylovových prostorů (krylovovských metod) užívaných při řešení systémů lineárních rovnic a odvozením jejich algoritmů. Krylovovská metoda je speciálním případem metody projektivní. Určuje aproximace řešení systému lineárních rovnic ve varietách tvořených počáteční aproximací řešení a prostory postupně rostoucích dimenzí (Krylovovy podprostory). Aproximace jsou přitom jednoznačně dány určujícími podmínkami kladenými na korespondující reziduový vektor. Podle volby určujících podmínek rozlišujeme několik základních krylovovských metod. Jejich algoritmické vyjádření není jednoznačné a závisí na druhu použité báze Krylovova prostoru (Arnoldiho, Lanczosova), na algoritmu, který vektory báze počítá a konečně na způsobu konstrukce požadované aproximace a korespondujícího rezidua z vektorů báze. Volba vhodných algoritmů realizujících jednotlivé fáze výpočtu je důležitá především v souvislosti s použitím algoritmů v konečné aritmetice počítače. Uvedeme čtyři základní metody Krylovových prostorů a budeme se zabývat jejich algoritmickým vyjádřením. Z algoritmu BiCG*, který používá Lanczosovu bázi, odvodíme algoritmy metod Lanczosova typu a vysvětlíme základní ideu hybridních* BiCG*-metod. V závěru kapitoly budeme diskutovat zastavovací kritérium algoritmů.*

Uvažujme systém lineárních rovnic

$$\mathbf{A}x = b, \tag{1.1}$$

kde $\mathbf{A} \in \mathbb{R}^{n\times n}$ je reálná regulární matice, $b \in \mathbb{R}^n$ vektor pravé strany a $x \in \mathbb{R}^n$ označuje řešení soustavy (1.1). Pro zjednodušení notace předpokládáme $\mathbf{A}$, b reálné; analogie všech postupů je možná i pro komplexní data.

Jedním z možných přístupů ke konstrukci aproximace řešení $\hat{x}$ soustavy (1.1) je hledat tuto aproximaci ve vhodném prostoru $\mathcal{K} \subset \mathbb{R}^n$ dimenze k resp. ve varietě $x_0 + \mathcal{K}$, kde x_0 je počáteční aproximace řešení, přičemž aproximace $\hat{x}$ je jednoznačně určena k určujícími podmínkami kladenými na příslušný reziduový vektor $b - \mathbf{A}\hat{x}$. Běžně užívané určující podmínky jsou ortogonalita rezidua na k lineárně nezávislých vektorů, jež společně určují prostor $\mathcal{L}$ dimenze k nazývaný *levý prostor*. Aproximace $\hat{x}$ je potom jednoznačně dána podmínkou

$$\hat{x} \in x_0 + \mathcal{K}, \quad b - \mathbf{A}\hat{x} \perp \mathcal{L}. \tag{1.2}$$

Popsané určující podmínky jsou běžně používány v mnoha různých matematických metodách a jsou známy jako *Petrov-Galerkinovy podmínky* a při volbě $\mathcal{L} \equiv \mathcal{K}$ jako *Galerkinovy podmínky*.

Projektivní metoda používá posloupnosti k-dimenzionálních prostorů $\mathcal{K}_k$ a $\mathcal{L}_k$. Aproximaci x_k poté určuje v každém kroku pomocí podmínky (1.2), přičemž za prostory $\mathcal{K}$ a $\mathcal{L}$ dosazuje prostory $\mathcal{K}_k$ a $\mathcal{L}_k$,

$$x_k \in x_0 + \mathcal{K}_k, \quad b - \mathbf{A}x_k \perp \mathcal{L}_k. \tag{1.3}$$

Z podmínky (1.3) plyne, že rostou-li dimenze prostorů $\mathcal{K}_k$ a $\mathcal{L}_k$, získáme nejvýše v n-tém kroku přesné řešení x soustavy (1.1).

Důležitou třídu projektivních metod pro řešení soustavy (1.1) tvoří *metody Krylovových prostorů* (zkráceně *krylovovské metody*), u nichž se za členy posloupnosti prostorů $\mathcal{K}_k$ volí Krylovovy podprostory $\mathcal{K}_k(\mathbf{A}, r_0)$,

$$\mathcal{K}_k(\mathbf{A}, r_0) \equiv \mathrm{span}\{r_0, \mathbf{A}r_0, \ldots, \mathbf{A}^{k-1}r_0\},$$

kde $r_0 \equiv b - \mathbf{A}x_0$ a k-tá aproximace řešení x_k,

$$x_k \in x_0 + \mathcal{K}_k(\mathbf{A}, r_0), \tag{1.4}$$

soustavy (1.1) je určena pomocí k podmínek aplikovaných na vektor $r_k \equiv b - \mathbf{A}x_k$. Pro reziduum r_k podle (1.4) platí

$$r_k \in r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0). \tag{1.5}$$

Určující podmínky pro reziduum nemusí být dány při formulaci metody explicitně jako ortogonální. I přesto lze krylovovské metody implicitně považovat za metody projektivní neboť libovolné určující podmínky lze převést na podmínku ortogonality rezidua k určitému prostoru.

Kapitola má následující strukturu. V sekci 1.1 přistoupíme k formulaci krylovovských metod a budeme diskutovat jejich základní vlastnosti plynoucí z určujících podmínek. Sekce 1.2 je věnována konstrukci a odvození různých algoritmů realizujících počítání báze Krylovova prostoru. V sekci 1.3 ukazujeme, jakým způsobem z bázových vektorů zkonstruovat aproximace řešení požadovaných vlastností a zabýváme se vztahem metod s minimálním a galerkinovským kvazi-reziduem. Na základě výsledků z 1.2 a 1.3 formulujeme v 1.4 algoritmy základních krylovovských metod GMRES, FOM, LM a QMR. V sekci 1.5 odvozujeme algoritmy metod Lanczosova typu a popisujeme základní myšlenku hybridních BiCG-metod, mezi které se řadí např. BiCGStab(l). Kapitolu zakončujeme diskusí o zastavovacím kritériu algoritmů.

1.1 Základní krylovovské metody

V úvodu této kapitoly jsme vysvětlili, že krylovovské metody lze implicitně chápat jako metody projektivní a zřejmě můžeme definovat různé krylovovské metody volbou levého prostoru. Důležité přitom je, abychom byli schopni vektory určené konkrétními podmínkami rozumně vyjádřit, t.j. nalézt algoritmus k jejich vypočtení. Na základě volby levého prostoru můžeme definovat následující běžně užívané metody:

FOM (*Full orthogonalization method*)

$$x_k \in x_0 + \mathcal{K}_k(\mathbf{A}, r_0), \quad r_k \perp \mathcal{K}_k(\mathbf{A}, r_0), \tag{1.6}$$

GMRES (*Generalized Mimimal Residual Method*)

$$x_k \in x_0 + \mathcal{K}_k(\mathbf{A}, r_0), \quad r_k \perp \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0), \tag{1.7}$$

LM (*Lanczos Method for solving linear systems*)

$$x_k \in x_0 + \mathcal{K}_k(\mathbf{A}, r_0), \quad r_k \perp \mathcal{K}_k(\mathbf{A}^T, \widetilde{r}_0), \tag{1.8}$$

kde $\widetilde{r}_0$ je pomocný nenulový vektor. Poznamenejme, že volby levých prostorů v (1.6), (1.7) a (1.8) jsou pouze jedny z možných.

Abychom mohli pracovat s Krylovovými prostory a vypočítávat vektory vyhovující různým podmínkám, bude zřejmě nutné zvolit bázi Krylovova podprostoru $\mathcal{K}_k(\mathbf{A}, r_0)$. Budeme uvažovat takové bázové vektory $v_1, \ldots, v_k$ podprostoru $\mathcal{K}_k(\mathbf{A}, r_0)$, které vyhovují podmínkám

$$v_i \in \mathcal{K}_i(\mathbf{A}, r_0) - \mathcal{K}_{i-1}(\mathbf{A}, r_0), \quad \text{span}(v_1, \ldots, v_i) = \mathcal{K}_i(\mathbf{A}, r_0) \tag{1.9}$$

a bez újmy na obecnosti předpokládejme, že $\|v_i\| = 1$, $1 \leq i \leq k$. Při volbě konkrétní báze je důležité abychom byli schopni proces vytváření báze dobře algoritmizovat. Příkladem bází vyhovujících této podmínce jsou *Arnoldiho* a *Lanczosova* báze. Vektory Arnoldiho báze jsou určeny podmínkou (1.9) a podmínkou

$$v_i^T v_j = \delta_{ij}, \tag{1.10}$$

kde δ_{ij} označuje Kronekerovo delta, t.j. $v_1, \ldots, v_k$ tvoří ortonormální bázi Krylovova podprostoru $\mathcal{K}_k(\mathbf{A}, r_0)$. Vektory Lanczosovy báze jsou dány podmínkami (1.9) a

$$v_i \perp \mathcal{K}_{i-1}(\mathbf{A}^T, \widetilde{r}_0), \quad 1 < i \leq k. \tag{1.11}$$

Označme $\mathbf{V}_k \in \mathbb{R}^{n \times k}$ matici, jejíž sloupce postupně tvoří vektory $v_1, \ldots, v_k$. Potom lze libovolný reziduový vektor r_{k-1} z variety $r_0 + \mathbf{A}\mathcal{K}_{k-1}(\mathbf{A}, r_0) \subset \mathcal{K}_k(\mathbf{A}, r_0)$ psát ve tvaru

$$r_{k-1} = \mathbf{V}_k \, q_{k-1}. \tag{1.12}$$

Vektor $q_{k-1} \in \mathbb{R}^k$ je zřejmě souřadnicový vektor rezidua r_{k-1} v bázi $v_1, \ldots, v_k$ a budeme ho nazývat *kvazi-reziduem*. Pokud je kvazi-reziduum voleno tak, aby byl reziduový vektor r_{k-1} násobkem posledního bázového vektoru v_k, budeme hovořit o *galerkinovském* kvazi-reziduu. Vektor q_{k-1} s minimální normou ze všech přípustných kvazi-reziduí (t.j. takových, že reziduum r_k určené (1.12) splňuje (1.5)) se nazývá *minimální* kvazi-reziduum. Povšimněme si následujících skutečností:

- Metoda s galerkinovskými kvazi-rezidui používající Arnoldiho bázi je metoda FOM. Reziduový vektor je totiž násobkem příslušného bázového vektoru Arnoldiho báze a nutně splňuje podmínku (1.6).
- Metodu s galerkinovskými kvazi-rezidui používající Lanczosovu bázi známe z (1.8) jako LM. Reziduum r_k je násobkem příslušného bázového vektoru splňujícího (1.11) pro $i = k + 1$ a r_k je nutně kolmé na $\mathcal{K}_k(\mathbf{A}^T, \widetilde{r}_0)$.
- Metoda s minimálními kvazi-rezidui používající Arnoldiho bázi je GMRES formulovaná podmínkami (1.7). Protože v případě Arnoldiho báze platí

  $$\|\mathbf{V}_{k+1} \, q_k\| = \|q_k\|$$

  a q_k má minimální normu ze všech přípustných kvazi-reziduí, má zřejmě reziduum $r_k = \mathbf{V}_{k+1} \, q_k$ nejmenší normu ze všech reziduí z variety $r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$, platí

  $$\|b - \mathbf{A}x_k\| = \min_{x \in x_0 + \mathcal{K}_k(\mathbf{A}, r_0)} \|b - \mathbf{A}x\|. \tag{1.13}$$

  Reziduum r_k splňující podmínku (1.13) nutně leží ve varietě $r_k \in r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$ a je kolmé na prostor $\mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$ (protože má minimální normu) a tedy splňuje podmínku (1.7).

Metodu s minimálními kvazi-rezidui používající Lanczosovu bázi budeme nazývat QMR (*Quasi Minimal Residual Method*). I tuto metodu lze implicitně považovat za projektivní metodu ve smyslu podmínek (1.3).

Právě uvedený postup formulace krylovovských metod (pomocí konkrétně zvolené báze a kvazi-rezidua) naznačuje, že pro určení reziduí a aproximací krylovovské metody bude nutné algoritmicky vyřešit dva problémy: konstrukci bázových vektorů a konstrukci reziduí a aproximací na základě znalosti vektorů báze.

Je-li matice $\mathbf{A}$ symetrická, jsou algoritmy metod určených podmínkami (1.6) a (1.7) laciné (co se týče počtu operací potřebných k provedení jedné iterace) a paměťově nenáročné. Metodu formulovanou podmínkou (1.6) potom nazýváme metodou sdružených gradientů (CG) [32]. Pokud je matice $\mathbf{A}$ pozitivně definitní, určuje metoda CG aproximace optimální ve smyslu minimální $\mathbf{A}$-normy chyby. K metodě CG se vrátíme v kapitole třetí, čtvrté a páté, v nichž budeme hovořit o odhadu $\mathbf{A}$-normy chyby, chování CG v konečné aritmetice a analýze zaokrouhlovacích chyb v odhadech $\mathbf{A}$-normy chyby. Metodu formulovanou pomocí podmínky (1.7) nazýváme v symetrickém případě metodou minimálních reziduí (MinRes). Laciný a paměťově nenáročný algoritmus této metody nám dává možnost vypočíst aproximace optimální ve smyslu minimálního rezidua.

Chceme-li aplikovat krylovovské metody v obecně nesymetrickém případě, stojíme před otázkou zda použít metody používající Lanczosovu nebo Arnoldiho bázi.

Metody používající Arnoldiho bázi (GMRES a FOM) jsou založeny na ortogonalizačních algoritmech, realizujících počítání ortonormální báze Krylovova prostoru. Abychom mohli vyjádřit další bázový vektor, je obecně nutné udržovat v paměti počítače všechny předchozí bázové vektory. Proto hovoříme o těchto metodách jako o metodách s dlouhými rekurentními vztahy (dále jen dlouhými rekurencemi). Udržování vektorů v paměti počítače samozřejmě vede s přibývajícími iteracemi k velkým nárokům na paměť počítače a na zvyšující se počet operací potřebných k provedení jedné iterace. Na druhé straně jsou tyto metody robustní a numericky stabilní (samozřejmě realizujeme-li danou metodu vhodným algoritmem). Metoda GMRES navíc počítá aproximace optimální ve smyslu minimálního rezidua (1.13). Mezi metodou GMRES a FOM existuje velmi úzký vztah, který popíšeme v oddíle 1.3.3.

Algoritmy metod používající Lanczosovu bázi (LM a QMR) mají tu výhodu, že k vypočtení dalšího bázového vektoru není nutné uchovávat všechny předchozí bázové vektory v paměti počítače, hovoříme o metodách s krátkými rekurencemi. Algoritmy metod LM a QMR jsou paměťově nenáročné a laciné ve smyslu počtu operací na jednu iteraci. U těchto metod nelze obecně hovořit o optimalitě počítaných aproximací v nějakém smyslu. Přesto jsou často konvergenční reziduové křivky metod používajících Lanczosovu bázi velmi blízké optimální křivce GMRES. Mezi základní nevýhody algoritmů realizujících metody LM a QMR patří možná numerická nestabilita. Podíly, z nichž se počítají koeficienty lineárních kombinací vektorů jsou často velmi velké a ztráta informace způsobená zaokrouhlovacími chybami může způsobit znehodnocení výpočtu algoritmu.

1.2 Báze Krylovových prostorů

Motorem každé krylovovské metody je algoritmus (používající klasické projektivní postupy), jež vytváří bázi Krylovova prostoru a předem tím většinou určuje, zda se bude jednat o metodu s dlouhými či krátkými rekurencemi. Vzniklou bázi poté můžeme využít ke konstrukci aproximace či rezidua požadovaných vlastností.

O vektorech báze má smysl hovořit do té doby, dokud rostou dimenze Krylovových podprostorů. Maximální dimenzi, jež mohou postupně rostoucí Krylovovy prostory generované maticí $\mathbf{A}$ a vektorem r_0 dosáhnout, budeme nazývat *stupněm* r_0 *vzhledem k* $\mathbf{A}$ [30]

a značit ji symbolem $\vartheta(\mathbf{A}, r_0)$. Platí vztah

$$\vartheta(\mathbf{A}, r_0) = \min\{k;\ \dim \mathcal{K}_k(\mathbf{A}, r_0) = \dim \mathcal{K}_{k+1}(\mathbf{A}, r_0)\}.$$

Pro libovolný vektor r_0 je velikost $\vartheta(\mathbf{A}, r_0)$ zřejmě omezena stupněm minimálního polynomu. Z trochu jiného pohledu můžeme číslo $\vartheta(\mathbf{A}, r_0)$ chápat jako dimenzi nejmenšího Krylovova podprostoru $\mathcal{K}_k(\mathbf{A}, r_0)$ invariantního vzhledem k násobení maticí $\mathbf{A}$ ($x \in \mathcal{K}_k(\mathbf{A}, r_0) \Rightarrow \mathbf{A}x \in \mathcal{K}_k(\mathbf{A}, r_0)$). Příkladem invariantních prostorů mohou být prostory tvořené z vlastních vektorů příslušných k reálnému vlastnímu číslu resp. reálných a imaginárních složek vlastních vektorů příslušných ke komplexnímu vlastnímu číslu či z řetězů zobecněných vlastních vektorů, případně jejich reálných a imaginárních částí.

Bázi Krylovova prostoru splňující podmínku (1.9) je nutné počítat vhodným způsobem, například tak, že nový bázový vektor v_{k+1} vypočteme jako lineární kombinaci předchozích bázových vektorů a posledního bázového vektoru na který aplikujeme matici $\mathbf{A}$. Uvedený postup lze popsat následující rekurencí

$$v_{k+1} = \frac{1}{h_{k+1k}} \left(\mathbf{A}v_k - \sum_{i=1}^{k} h_{ik} v_i \right), \tag{1.14}$$

kde h_{ik} jsou vhodně zvolené koeficienty a $h_{k+1k} \neq 0$ lze použít pro určení délky daného bázového vektoru. Volba rekurence (1.14) není očividně omezující. Splňují-li totiž vektory $v_1, \ldots, v_k$ podmínku (1.9), je možno rekurencí (1.14) vyjádřit libovolný vektor z prostoru $\mathcal{K}_{k+1}(\mathbf{A}, r_0)$. Navíc platí, že vektory $v_1, \ldots, v_k, v_{k+1}$ splňují opět podmínku (1.9) s indexem o jedničku vyšším, je-li $k < \vartheta(\mathbf{A}, r_0)$. Rekurence (1.14) dává do značné míry univerzální návod na počítání lineárně nezávislých bázových vektorů splňujících podmínku (1.9).

Označme $\mathbf{V}_k \in \mathbb{R}^{n \times k}$ matici, jejíž sloupce postupně tvoří vektory $v_1, \ldots, v_k$,

$$\mathbf{V}_k \equiv (v_1, \ldots, v_k).$$

Způsob konstrukce báze poté můžeme vzhledem k (1.14) vyjádřit maticovou rovností

$$\mathbf{A}\mathbf{V}_k = \mathbf{V}_{k+1} \underline{\mathbf{H}}_k, \tag{1.15}$$

kde $\underline{\mathbf{H}}_k \in \mathbf{R}^{(k+1)\times k}$ je matice ve tvaru

$$\underline{\mathbf{H}}_k \equiv \begin{pmatrix} h_{11} & h_{12} & h_{13} & \ldots & h_{1k} \\ h_{21} & h_{22} & h_{23} & \ldots & h_{2k} \\ 0 & h_{32} & h_{33} & \ldots & h_{3k} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ldots & 0 & \ddots & h_{kk} \\ 0 & \ldots & 0 & 0 & h_{k+1k} \end{pmatrix}.$$

Z (1.14) je patrné, že matice $\underline{\mathbf{H}}_k$ vznikla z matice $\underline{\mathbf{H}}_{k-1}$ přidáním posledního sloupce.

Pro další použití označme $\mathbf{H}_k \in \mathbb{R}^{k\times k}$ horní Hessenbergovu matici, jež vznikne z $\underline{\mathbf{H}}_k$ vynecháním posledního řádku. Rovnost (1.15) potom můžeme psát ve tvaru

$$\mathbf{A}\mathbf{V}_k = \mathbf{V}_k \mathbf{H}_k + h_{k+1k} v_{k+1} e_k^T, \tag{1.16}$$

kde $e_k \in \mathbb{R}^k$ označuje k-tý sloupec matice identity, $e_k = (0, \ldots, 0, 1)^T$. Vztah (1.16) vyjadřuje závislost mezi bázovými vektory. Pokud bude mít matice $\mathbf{H}_k$ zaplněný horní trojúhelník nenulovými prvky, nebude zřejmě možné sestrojit další bázový vektor bez pomoci obecně všech předchozích bázových vektorů. Jestliže se nám však podaří docílit toho, aby byla matice $\mathbf{H}_k$ například třídiagonální, bude možné vypočíst další bázový vektor pouze ze znalosti předchozích dvou vektorů báze. V následujících odstavcích odvodíme z rekurence (1.14) známé algoritmy počítající Arnoldiho a Lanczosovu bázi.

1.2.1 Arnoldiho báze

Pro každé $k \leq \vartheta(\mathbf{A}, r_0)$ existuje posloupnost vzájemně ortogonálních vektorů $v_1, \ldots, v_k$ s euklidovskou normou rovnou jedné taková, že platí $\mathcal{K}_k(\mathbf{A}, r_0) = \mathrm{span}(v_1, \ldots, v_k)$. Ortonormální bázi Krylovova prostoru budeme označovat jako *Arnoldiho bázi* a proces konstrukce Arnoldiho báze jako *Arnoldiho proces*.

Předpokládejme, že vektory $v_1, \ldots, v_k$ tvoří ortonormální bázi Krylovova podprostoru $\mathcal{K}_k(\mathbf{A}, r_0)$. Následující bázový vektor v_{k+1} lze počítat rekurencí

$$\mathbf{t} = \mathbf{A}v_k - \sum_{i=1}^{k} (v_i, \mathbf{A}v_k)\, v_i, \quad h_{kk+1} = \|\mathbf{t}\|, \quad v_{k+1} = h_{kk+1}^{-1}\mathbf{t}. \tag{1.17}$$

Rekurence (1.17) je zřejmě rekurence (1.14) s volbou koeficientů $h_{ik} = (v_i, \mathbf{A}v_k)$ a koeficient h_{kk+1} je určen tak, aby platilo $\|v_{k+1}\| = 1$. Algoritmus, který počítá vektory ortonormální báze Krylovova prostoru podle (1.17) budeme nazývat *klasický Arnoldiho algoritmus*. Jde vlastně o klasický Gram-Schmidtův algoritmus upravený pro počítání ortonormální báze Krylovova prostoru.

Matematicky ekvivalentní formou klasického Arnoldiho algoritmu je modifikovaný Arnoldiho algoritmus 1.1, který je vhodnější při počítání v aritmetice s konečnou přesností.

Algoritmus 1.1. *Modifikovaný Arnoldiho algoritmus*

input $x_0,\ \mathbf{A},\ b$

initialization

$$\begin{aligned} r_0 &= b - \mathbf{A}x_0 \\ v_1 &= r_0/\|r_0\| \end{aligned}$$

for $j = 1, \ldots, k-1$

$$\mathbf{t} = \mathbf{A}v_j$$

for $i = 1, \ldots, j$

$$\begin{aligned} h_{ij} &= v_i^T \mathbf{t} \\ \mathbf{t} &= \mathbf{t} - h_{ij} v_i \end{aligned}$$

end for

$$\begin{aligned} h_{j+1j} &= \|\mathbf{t}\| \\ v_{j+1} &= \mathbf{t}/h_{j+1j} \end{aligned}$$

end for

Myšlenka klasického i modifikovaného Arnoldiho algoritmu spočívá v tom, že se od vektoru $\mathbf{A}v_j \in \mathcal{K}_{j+1}(\mathbf{A}, r_0)$ odečítá jeho projekce do prostoru $\mathcal{K}_j(\mathbf{A}, r_0)$ a tím získáme vektor ležící v $\mathcal{K}_{j+1}(\mathbf{A}, r_0)$ kolmý na $\mathcal{K}_j(\mathbf{A}, r_0)$, tedy i na všechny vektory $v_1, \ldots, v_j$. Normováním tohoto vektoru dostáváme vektor v_{j+1}. Jak je patrno, rozdíl mezi oběma algoritmy je pouze ve způsobu výpočtu koeficientů h_{ij}. V případě klasického Arnoldiho algoritmu odečítáme od vektoru $\mathbf{A}v_j$ postupně jeho projekce do prostorů generovaných jednotlivými vektory v_i. V modifikovaném potom odečítáme od vektoru $\mathbf{A}v_j$ projekce nikoliv přímo tohoto vektoru, ale postupně počítaných vektorů $\mathbf{t}$.

Arnoldiho algoritmus zřejmě určuje ortonormální vektory a k jeho ukončení dochází tehdy, je-li $h_{j+1j} = 0$. Pokud však je $h_{j+1j} = 0$, potom z rekurence (1.14) plyne

$$\mathbf{A}v_j = \sum_{i=1}^{j} h_{ij} v_i,$$

vektor $\mathbf{A}v_j$ zřejmě leží v prostoru $\mathcal{K}_j(\mathbf{A}, r_0)$, platí

$$\mathcal{K}_j(\mathbf{A}, r_0) = \text{span}\{v_1, \ldots, v_j\} = \text{span}\{v_1, \ldots, v_j, \mathbf{A}v_j\} = \mathcal{K}_{j+1}(\mathbf{A}, r_0)$$

a tudíž je $j = \vartheta(\mathbf{A}, r_0)$.

Ke konstrukci Aroldiho báze je možné použít i jiných postupů než rekurence (1.14), jak ukázal Walker ve své práci [65]. Volíme-li vektor v_1 jako normovaný vektor r_0, je nový bázový vektor v_{k+1} dán $(k+1)$-ním sloupcem matice řádu $n \times n$, která je součinem elementárních Hauseholderových matic, podrobněji, elementární Hauseholderovu matici $\mathbf{P}_{k+1}$ $(k \geq 0)$ volíme tak, že

$$\mathbf{P}_{k+1}\mathbf{P}_k \ldots \mathbf{P}_1(v_1, \mathbf{A}\mathbf{V}_k) = \begin{pmatrix} \mathbf{R}_{k+1} \\ 0 \end{pmatrix},$$

kde $\mathbf{R}_{k+1}$ je horní trojúhelníková $(k+1) \times (k+1)$ matice a odtud

$$\left(v_1, \mathbf{A}\mathbf{V}_k\right) = \mathbf{V}_{k+1}\mathbf{R}_{k+1}, \quad \mathbf{V}_{k+1} \equiv \mathbf{P}_{k+1}\mathbf{P}_k \ldots \mathbf{P}_1 \begin{pmatrix} \mathbf{I}_{k+1} \\ 0 \end{pmatrix}, \tag{1.18}$$

$\mathbf{I}_{k+1} \in \mathbb{R}^{(k+1)\times(k+1)}$ je matice identity a sloupce $\mathbf{V}_{k+1}$ tvoří ortonormální bázi prostoru $\mathcal{K}_{k+1}(\mathbf{A}, r_0)$. Poznamenejme, že vzhledem k (1.15) a (1.18) platí $\mathbf{R}_{k+1} = \left(e_1, \underline{\mathbf{H}}_k\right)$.

1.2.2 Lanczosova báze

V tomto odstavci se budeme zabývat odvozením algoritmů, podle nichž lze vypočíst vektory Lanczosovy báze určené pro $k > 1$ podmínkou

$$v_{k+1} \in \mathcal{K}_{k+1}(\mathbf{A}, r_0) - \mathcal{K}_k(\mathbf{A}, r_0), \quad v_{k+1} \perp \mathcal{K}_k(\mathbf{A}^T, \widetilde{r}_0), \quad \|v_{k+1}\| = 1. \tag{1.19}$$

Vektor $\widetilde{r}_0$ je pomocný nenulový vektor takový, že $\widetilde{r}_0^T r_0 \neq 0$ a bývá nazýván *levým startovacím vektorem* nebo též *stínovým vektorem* [30]. Jeho zkoumání se budeme věnovat v kapitole 2.

Při odvozování algoritmu vyjdeme opět z rekurence (1.14) a budeme se snažit určit koeficienty h_{ik} tak, aby výsledný vektor vypočtený rekurencí (1.14) splňoval podmínku (1.19). Protože hledáme vektor v_{k+1} kolmý ke Krylovovu podprostoru $\mathcal{K}_k(\mathbf{A}^T, \widetilde{r}_0)$, je zřejmě vhodné určit bázi tohoto podprostoru, vektory $w_1, \ldots, w_k$. Definujme tyto vektory podmínkou analogickou k (1.19), t.j.

$$w_{j+1} \in \mathcal{K}_{j+1}(\mathbf{A}^T, \widetilde{r}_0) - \mathcal{K}_j(\mathbf{A}^T, \widetilde{r}_0), \quad w_{j+1} \perp \mathcal{K}_j(\mathbf{A}, r_0), \quad 1 \leq j < k \tag{1.20}$$

a normujme je například tak, že platí

$$w_j^T v_j = 1, \quad 1 \leq j \leq k. \tag{1.21}$$

Mezi bázovými vektory $w_1, \ldots, w_k$ a $v_1, \ldots, v_k$ zřejmě platí vztah

$$w_i^T v_j = \delta_{ij}, \tag{1.22}$$

kde δ_{ij} je Kronekerovo delta. Posloupnosti vektorů $\{v_j\}_{j=1}^k$ a $\{w_j\}_{j=1}^k$ splňující podmínku (1.22) nazveme *biortogonální*.

Bázové vektory w_j budeme počítat podle rekurence analogické k (1.14)

$$w_{k+1} = \frac{1}{g_{k+1k}}\left(\mathbf{A}^T w_k - \sum_{i=1}^{k} g_{ik} w_i\right) \tag{1.23}$$

a vhodnou volbou koeficientů g_{ik} a g_{k+1k} se budeme snažit o splnění podmínek (1.20) a (1.21). Uvažujeme-li vektor v_{k+1} ve tvaru (1.14), lze rovnice $w_i^T v_{k+1} = 0$, $1 \leq i \leq k$, psát s využitím biortogonálních podmínek (1.22) ve tvaru

$$0 = w_i^T \mathbf{A} v_k - h_{ik}\, w_i^T v_i \tag{1.24}$$

a podobně, uvažujeme-li w_{k+1} ve tvaru (1.23), lze zapsat rovnice $w_{k+1}^T v_i = 0$ jako

$$0 = w_k^T \mathbf{A} v_i - g_{ik}\, w_i^T v_i. \tag{1.25}$$

Uvědomme si, že pro $i \leq k-2$ platí $w_i^T \mathbf{A} v_k = 0$ a z (1.21), (1.24) a (1.25) potom plyne

$$h_{ik} = 0 = g_{ik}. \tag{1.26}$$

Dosadíme-li do (1.24) a (1.25) za index i postupně $k-1$ a k, dostáváme s využitím (1.21)

$$h_{k-1k} = w_{k-1}^T \mathbf{A} v_k, \tag{1.27}$$

$$g_{k-1k} = w_k^T \mathbf{A} v_{k-1} \tag{1.28}$$

a

$$h_{kk} = w_k^T \mathbf{A} v_k = g_{kk}. \tag{1.29}$$

Koeficient h_{k+1k} je určen podmínkou $\|v_{k+1}\| = 1$. Dosadíme-li do rovnice $v_{k+1}^T w_{k+1} = 1$ vektor w_{k+1} ve tvaru (1.23) a využijeme-li biortogonálních podmínek, dostáváme

$$g_{k+1k} = w_k^T \mathbf{A} v_{k+1} \tag{1.30}$$

a vektor w_{k+1} lze zřejmě vypočíst jen tehdy, je-li $g_{k+1k} \neq 0$. Podobně, dosazením vektoru v_{k+1} ve tvaru (1.14) do rovnice $v_{k+1}^T w_{k+1} = 1$ získáme vyjádření koeficientu h_{k+1k}

$$h_{k+1k} = w_{k+1}^T \mathbf{A} v_k. \tag{1.31}$$

V souladu s klasickým označením definujme koeficienty α_k, β_{k-1}, $\widetilde{\beta}_{k-1}$ γ_k, $\widetilde{\gamma}_k$ následujícím způsobem

$$\alpha_k \equiv h_{kk}, \qquad \beta_{k-1} \equiv h_{k-1k}, \qquad \gamma_k \equiv h_{k+1k}, \tag{1.32}$$

$$\widetilde{\beta}_{k-1} \equiv g_{k-1k}, \qquad \widetilde{\gamma}_k \equiv g_{k+1k}. \tag{1.33}$$

Ze vztahů (1.27)–(1.31) plyne

$$\gamma_k = \widetilde{\beta}_k, \qquad \widetilde{\gamma}_k = \beta_k. \tag{1.34}$$

Na základě rekurencí (1.14) a (1.23), určení koeficientů podle (1.26)–(1.31), označení (1.32)–(1.33) a vztahů (1.34) lze formulovat algoritmus 1.2 pro počítání Lanczosovy báze, který nazýváme *Nesymetrický Lanczosův algoritmus* (NL).

K ukončení algoritmu 1.2 dochází tehdy, jestliže platí

$$\gamma_k = 0 \qquad \text{nebo} \qquad \beta_k = 0. \tag{1.35}$$

Podle příčin, které vedou ke splnění (1.35) rozlišujeme dva způsoby ukončení nesymetrického Lanczosova algoritmu viz. [23].

Algoritmus 1.2. *Nesymetrický Lanczosův algoritmus* (NL)

input $\mathbf{A}$, r_0, $\widetilde{r}_0$

initialization

$$
\begin{aligned}
v_0 &= \mathbf{o} \\
w_0 &= \mathbf{o} \\
\beta_0 &= 0 \\
\gamma_0 &= 0 \\
v_1 &= r_0/\|r_0\| \\
w_1 &= \widetilde{r}_0/\widetilde{r}_0^T v_1
\end{aligned}
$$

for $k = 1, \ldots$

$$
\begin{aligned}
\alpha_k &= w_k^T \mathbf{A} v_k \\
\mathbf{t} &= \mathbf{A}v_k - \alpha_k v_k - \beta_{k-1} v_{k-1} \\
\gamma_k &= \|\mathbf{t}\| \\
v_{k+1} &= \mathbf{t}/\gamma_k \\
\mathbf{t} &= \mathbf{A}^T w_k - \alpha_k w_k - \gamma_{k-1} w_{k-1} \\
\beta_k &= v_{k+1}^T \mathbf{t} \\
w_{k+1} &= \mathbf{t}/\beta_k
\end{aligned}
$$

end for

1. Nastane-li situace

$$\mathbf{o} = \mathbf{A}v_k - \alpha_k v_k - \beta_{k-1} v_{k-1}$$

nebo

$$\mathbf{o} = \mathbf{A}^T w_k - \alpha_k w_k - \gamma_{k-1} w_{k-1},$$

jsou vektory ve výše popsaných rekurencích lineárně závislé a pravý nebo levý Krylovův podprostor dosáhl své maximální dimenze, platí $k = \vartheta(\mathbf{A}, r_0)$ nebo $k = \vartheta(\mathbf{A}^T, \widetilde{r}_0)$. Tento způsob ukončení nazýváme „*obvyklé ukončení*“ (*regular termination*) [23].

2. Je-li směr netriviálního vektoru

$$\mathbf{A}^T w_k - \alpha_k w_k - \gamma_{k-1} w_{k-1}$$

kolmý na vektor v_{k+1}, dochází k *Lanczosovu ukončení* (*Lanczos breakdown*) a není možné vypočíst vektor w_{k+1}. Tomuto typu ukončení se můžeme vyhnout pomocí algoritmů používajících *look-ahead* strategie, o jejichž rozvoj a algoritmizaci se zasloužili např. R. W. Freund, M. H. Gutknecht a N. M. Nachtigal [13] nebo autoři práce [6]. Základní myšlenkou look-ahead postupů je konstruovat pouze takové vektory v_k a w_k, jejichž výpočet je stabilní. Kromě vektorů, při jejichž výpočtu by došlo k Lanczosovu ukončení se „přeskakují“ například i vektory, při jejichž výpočtu by došlo k numerickým nestabilitám a v algoritmu by se objevila příliš velká čísla způsobující znehodnocení výpočtu (*near breakdown*). Algoritmy používající look-ahead postupy samozřejmě vyžadují navíc pomocné vektory, jejichž počet je přímo úměrný délce povoleného skoku.

Obecně lze zřejmě počítat bázové vektory v_k a w_k na základě (1.14), (1.23)–(1.26) a (1.32)–(1.34) podle rekurencí

$$v_{k+1} = \gamma_k^{-1}(\mathbf{A}^T v_k - \alpha_k v_k - \beta_{k-1} v_{k-1}), \tag{1.36}$$

$$w_{k+1} = \widetilde{\gamma}_k^{-1}(\mathbf{A}^T w_k - \alpha_k w_k - \widetilde{\beta}_{k-1} w_{k-1}). \tag{1.37}$$

Nenulové koeficienty γ_k a $\widetilde{\gamma}_k$ určují délku počítaných vektorů a koeficienty α_k, β_k, $\widetilde{\beta}_k$ vypočteme na základě (1.24) a (1.25),

$$\alpha_k = \frac{w_k^T \mathbf{A} v_k}{w_k^T v_k}, \quad \beta_{k-1} = \frac{w_{k-1}^T \mathbf{A} v_k}{w_{k-1}^T v_{k-1}}, \quad \widetilde{\beta}_{k-1} = \frac{w_k^T \mathbf{A} v_{k-1}}{w_{k-1}^T v_{k-1}}. \tag{1.38}$$

Rekurence (1.36) a (1.37) lze zapsat v maticové podobě (1.16), platí

$$\mathbf{A}\mathbf{V}_k = \mathbf{V}_k \mathbf{T}_k + \gamma_k v_{k+1} e_k^T, \tag{1.39}$$

$$\mathbf{A}^T \mathbf{W}_k = \mathbf{W}_k \widetilde{\mathbf{T}}_k + \widetilde{\gamma}_k w_{k+1} e_k^T, \tag{1.40}$$

kde sloupce matice $\mathbf{V}_k$ jsou tvořeny postupně vektory $v_1, \ldots, v_k$, sloupce $\mathbf{W}_k$ vektory $w_1, \ldots, w_k$ a tří-diagonální matice $\mathbf{T}_k$ a $\widetilde{\mathbf{T}}_k$ mají tvar

$$\mathbf{T}_k \equiv \begin{pmatrix} \alpha_1 & \beta_1 & 0 & \ldots & 0 \\ \gamma_1 & \alpha_2 & \beta_2 & \ddots & \vdots \\ 0 & \gamma_2 & \alpha_3 & \ddots & 0 \\ \vdots & \ddots & \gamma_3 & \ddots & \beta_{k-1} \\ 0 & \ldots & 0 & \ddots & \alpha_k \end{pmatrix}, \quad \widetilde{\mathbf{T}}_k \equiv \begin{pmatrix} \alpha_1 & \widetilde{\beta}_1 & 0 & \ldots & 0 \\ \widetilde{\gamma}_1 & \alpha_2 & \widetilde{\beta}_2 & \ddots & \vdots \\ 0 & \widetilde{\gamma}_2 & \alpha_3 & \ddots & 0 \\ \vdots & \ddots & \widetilde{\gamma}_3 & \ddots & \widetilde{\beta}_{k-1} \\ 0 & \ldots & 0 & \ddots & \alpha_k \end{pmatrix}. \tag{1.41}$$

Aplikujeme-li na maticovou rovnost (1.39) zleva matici $\mathbf{W}_k^T$ a podobně, násobíme-li transponovanou maticovou rovnost (1.40) maticí $\mathbf{V}_k$ zprava, obdržíme maticové rovnosti

$$\mathbf{W}_k^T \mathbf{A} \mathbf{V}_k = \mathbf{W}_k^T \mathbf{V}_k \mathbf{T}_k + \gamma_k \mathbf{W}_k^T v_{k+1} e_k^T = \mathbf{D}_k \mathbf{T}_k, \tag{1.42}$$

$$\mathbf{W}_k^T \mathbf{A} \mathbf{V}_k = \widetilde{\mathbf{T}}_k^T \mathbf{W}_k^T \mathbf{V}_k + \widetilde{\gamma}_k e_k w_{k+1}^T \mathbf{V}_k = \widetilde{\mathbf{T}}_k^T \mathbf{D}_k. \tag{1.43}$$

kde $\mathbf{D}_k$

$$\mathbf{D}_k \equiv \mathbf{W}_k^T \mathbf{V}_k$$

je diagonální matice řádu k, $\mathbf{D}_k = \mathrm{diag}(\delta_1, \ldots, \delta_k)$, $\delta_i \equiv w_i^T v_i$, $i = 1, \ldots, k$. V rovnostech (1.42) a (1.43) jsme použili biortogonality vektorů. Z (1.42) a (1.43) plyne

$$\mathbf{D}_k \mathbf{T}_k = \widetilde{\mathbf{T}}_k^T \mathbf{D}_k.$$

Srovnání pod- a nad-diagonálních prvků vede k rovnostem

$$\delta_i \beta_i = \delta_{i+1} \widetilde{\gamma}_i, \quad \delta_{i+1} \gamma_i = \delta_i \widetilde{\beta}_i.$$

Vynásobíme-li první rovnost γ_i, druhou $\widetilde{\gamma}_i$ a srovnáme-li stejné výrazy, je

$$\gamma_i \beta_i = \widetilde{\beta}_i \widetilde{\gamma}_i. \tag{1.44}$$

Nenulové koeficienty γ_i a $\widetilde{\gamma}_i$ je zřejmě možné volit libovolným způsobem. Rovnosti (1.44) potom můžeme použít k určení koeficientů β_i a $\widetilde{\beta}_i$. Například v NL algoritmu 1.2 jsme použili volby $\gamma_i = \widetilde{\beta}_i$, $\widetilde{\gamma}_i = \beta_i$, viz. (1.34). V případě NL algoritmu 1.2 platí mezi maticemi $\widetilde{\mathbf{T}}_k$ a $\mathbf{T}_k$ vztah

$$\widetilde{\mathbf{T}}_k = \mathbf{T}_k^T. \tag{1.45}$$

Jiná, často používaná volba je

$$\gamma_i = \widetilde{\gamma}_i \tag{1.46}$$

a z (1.44) potom plyne $\beta_i = \widetilde{\beta}_i$, přičemž nenulový parametr γ_i můžeme volit libovolným způsobem. Uvažujeme-li volbu (1.46), platí

$$\widetilde{\mathbf{T}}_k = \mathbf{T}_k.$$

Použijme nyní volby (1.46) a maticové formy (1.39) rekurence (1.36) k odvození druhého algoritmu, jež rovněž počítá Lanczosovu bázi Krylovova prostoru. Předpokládejme, že existuje **LU**-rozklad matice $\mathbf{T}_k$,

$$\mathbf{T}_k = \mathbf{L}_k \mathbf{U}_k, \tag{1.47}$$

který lze rozepsáním matic $\mathbf{T}_k$, $\mathbf{L}_k$ a $\mathbf{U}_k$ po prvcích vyjádřit ve tvaru

$$\begin{pmatrix} \alpha_1 & \beta_1 & & \\ \gamma_1 & \alpha_2 & \ddots & \\ & \ddots & \ddots & \beta_{k-1} \\ & & \gamma_{k-1} & \alpha_k \end{pmatrix} = \begin{pmatrix} \varphi_1 & & & \\ \gamma_1 & \varphi_2 & & \\ & \ddots & \ddots & \\ & & \gamma_{k-1} & \varphi_k \end{pmatrix} \begin{pmatrix} 1 & \psi_1 & & \\ & 1 & \ddots & \\ & & \ddots & \psi_{k-1} \\ & & & 1 \end{pmatrix}. \tag{1.48}$$

Za pomoci (1.47) lze psát vztahy (1.39) a (1.40) (uvažujeme volbu (1.46)) ve tvaru

$$\mathbf{A}\mathbf{V}_k = \mathbf{V}_k \mathbf{L}_k \mathbf{U}_k + \gamma_k v_{k+1} e_k^T, \tag{1.49}$$

$$\mathbf{A}^T \mathbf{W}_k = \mathbf{W}_k \mathbf{L}_k \mathbf{U}_k + \gamma_k w_{k+1} e_k^T. \tag{1.50}$$

Definujme matice $\mathbf{P}_k^V$ a $\mathbf{P}_k^W$ vztahy

$$\mathbf{P}_k^V \equiv \mathbf{V}_k \mathbf{U}_k^{-1}, \qquad \mathbf{P}_k^W \equiv \mathbf{W}_k \mathbf{U}_k^{-1}. \tag{1.51}$$

Přenásobíme-li maticové rovnosti (1.49) a (1.50) zprava maticí $\mathbf{U}_k^{-1}$, užijeme-li definice (1.51) a skutečnosti, že platí $e_k^T \mathbf{U}_k^{-1} = e_k^T$ (na diagonále $\mathbf{U}_k^{-1}$ jsou jedničky), dostáváme

$$\mathbf{A}\mathbf{P}_k^V = \mathbf{V}_k \mathbf{L}_k + \gamma_k v_{k+1} e_k^T, \tag{1.52}$$

$$\mathbf{A}^T \mathbf{P}_k^W = \mathbf{W}_k \mathbf{L}_k + \gamma_k w_{k+1} e_k^T. \tag{1.53}$$

Zbývá pouze „přečíst“ algoritmus zapsaný v maticových rekurencích (1.51)–(1.53). Označme sloupce matice $\mathbf{P}_k^V$ jako $p_1^V, \ldots, p_k^V$ a sloupce matice $\mathbf{P}_k^W$ jako $p_1^W, \ldots, p_k^W$. Z (1.51) plyne $\mathbf{V}_k = \mathbf{P}_k^V \mathbf{U}_k$, $\mathbf{W}_k = \mathbf{P}_k^W \mathbf{U}_k$ a pro $i < k$ platí

$$v_{i+1} = p_{i+1}^V + \psi_i p_i^V, \tag{1.54}$$

$$w_{i+1} = p_{i+1}^W + \psi_i p_i^W. \tag{1.55}$$

Z (1.52) a (1.53) dostáváme

$$\mathbf{A} p_i^V = \varphi_i \, v_i + \gamma_i \, v_{i+1}, \tag{1.56}$$

$$\mathbf{A}^T p_i^W = \varphi_i \, w_i + \gamma_i \, w_{i+1}. \tag{1.57}$$

Dříve než odvodíme způsob jak vypočítat koeficienty φ_i a ψ_i, uvědomme si, jaký vztah platí mezi sloupci matic $\mathbf{P}_k^V$ a $\mathbf{P}_k^W$. Ze vztahu $\mathbf{V}_k = \mathbf{P}_k^V \mathbf{U}_k$ je zřejmé, že vektory $p_1^V, \ldots, p_k^V$ tvoří bázi prostoru $\mathcal{K}_k(\mathbf{A}, r_0)$; proto platí $w_{k+1}^T \mathbf{P}_k^V = 0$ a podobně $(\mathbf{P}_k^W)^T v_{k+1} = 0$ pro $i \leq k$. Násobíme-li maticovou rovnost (1.52) maticí $(\mathbf{P}_k^W)^T$ zleva a transpozici rovnosti (1.53) maticí $\mathbf{P}_k^V$ zprava, dostáváme

$$(\mathbf{P}_k^W)^T \mathbf{A} \mathbf{P}_k^V = (\mathbf{P}_k^W)^T \mathbf{V}_k \mathbf{L}_k, \tag{1.58}$$

$$(\mathbf{P}_k^W)^T \mathbf{A} \mathbf{P}_k^V = \mathbf{L}_k^T \mathbf{W}_k^T \mathbf{P}_k^V. \tag{1.59}$$

Protože platí $(p_i^W)^T v_j = 0$ pro $i < j$, je matice $(\mathbf{P}_k^W)^T \mathbf{V}_k$ dolní trojúhelníková a matice $(\mathbf{P}_k^W)^T \mathbf{V}_k \mathbf{L}_k$ je rovněž dolní trojúhelníková. Podobně lze ukázat, že matice $\mathbf{L}_k^T \mathbf{W}_k^T \mathbf{P}_k^V$ je horní trojúhelníková. Jelikož si jsou matice $(\mathbf{P}_k^W)^T \mathbf{V}_k \mathbf{L}_k$ a $\mathbf{L}_k^T \mathbf{W}_k^T \mathbf{P}_k^V$ podle (1.58) a (1.59) rovny, je matice $(\mathbf{P}_k^W)^T \mathbf{A} \mathbf{P}_k^V$ diagonální a zřejmě tedy platí

$$(p_i^W)^T \mathbf{A} p_j^V = 0, \quad i \neq j. \tag{1.60}$$

Říkáme, že vektory $\{p_i^V\}_{i=1}^k$ a $\{p_i^W\}_{i=0}^k$ jsou *bi-sdružené* či *bikonjugované*. Tvary koeficientů φ_i a ψ_i nyní již lehce odvodíme z rekurencí (1.54)–(1.57) s využitím biortogonálních a bikonjugovaných vztahů mezi vektory.

Uvědomme si nejdříve, že podle (1.55) je

$$w_i^T \mathbf{A} p_i^V = (p_i^W)^T \mathbf{A} p_i^V + \psi_{i-1} (p_{i-1}^W)^T \mathbf{A} p_i^V = (p_i^W)^T \mathbf{A} p_i^V,$$

přičemž v poslední rovnosti jsme použili (1.60). Přenásobíme-li (1.56) vektorem w_i^T, dostáváme

$$\varphi_i = \frac{w_i^T \mathbf{A} p_i^V}{w_i^T v_i} = \frac{(p_i^W)^T \mathbf{A} p_i^V}{w_i^T v_i}. \tag{1.61}$$

V odvození tvaru koeficientu ψ_i použijeme dva další vztahy plynoucí z rekurencí (1.54)–(1.57) a vztahů (1.22) a (1.60). Z rekurence (1.54) s indexem o jedničku nižším plyne, že platí

$$w_i^T v_i = w_i^T p_i^V + \psi_i w_i^T p_{i-1}^V = w_i^T p_i^V. \tag{1.62}$$

Dále, vyjádříme-li si z rekurence (1.57) vektor w_i a dosadíme-li ho do skalárním součinu $w_i^T p_{i+1}^V$, dostáváme

$$w_i^T p_{i+1}^V = \varphi_i^{-1} (\mathbf{A}^T p_i^W - \gamma_i w_{i+1})^T p_{i+1}^V = -\frac{\gamma_i}{\varphi_i} w_{i+1}^T v_{i+1}. \tag{1.63}$$

Přenásobíme-li nyní vztah (1.54) vektorem w_i^T a užijeme-li (1.62) a (1.63), dostáváme

$$\psi_i = -\frac{w_i^T p_{i+1}^V}{w_i^T p_i^V} = \frac{\gamma_i}{\varphi_i} \frac{w_{i+1}^T v_{i+1}}{w_i^T v_i}. \tag{1.64}$$

Na základě vztahů (1.54)–(1.57) a tvaru koeficientů (1.61) a (1.64) formulujeme algoritmus 1.3. Koeficient γ_k přitom volíme tak, aby platilo $\|v_{k+1}\| = 1$. Protože algoritmus užívá k určení vektorů Lanczosovy báze bikonjugovaných vektorů, pojmenovali jsme ho *Lanczosův algoritmus bikonjugovaných vektorů* (BiCV) [30]. K ukončení algoritmu dochází ve stejných případech jako v případě algoritmu NL, t.j. buď přestane růst dimenze jednoho z Krylovových podprostorů (obvyklé ukončení) a nebo je-li $v_k \neq \mathbf{o}$, $w_k \neq \mathbf{o}$ a $w_k^T v_k = 0$ (Lanczosovo ukončení). Navíc však může nastat situace taková, že neexistuje **LU**-rozklad matice $\mathbf{T}_k$ (1.48) a algoritmus je nutné ukončit, protože nelze dále počítat vektory bikonjugované báze. V algoritmu se tato situace projeví tím, že platí $(p_k^W)^T \mathbf{A} p_k^V = 0$, $\varphi_k = 0$ a není možné vypočíst koeficient ψ_k. I přes tuto nevýhodu je BiCV vhodnější při počítání v konečné aritmetice než NL [31]. Poznamenejme ještě, že z (1.48) plynou vztahy

$$\alpha_k = \varphi_k + \gamma_{k-1} \psi_{k-1}, \tag{1.65}$$

$$\beta_{k-1} = \varphi_{k-1} \psi_{k-1}. \tag{1.66}$$

ALGORITMUS 1.3. *Lanczosův algoritmus bikonjugovaných vektorů* (BiCV)

input $\mathbf{A}$, r_0, $\widetilde{r}_0$

initialization

$$\begin{aligned}
\gamma_0 &= \|r_0\| \\
v_1 &= r_0/\gamma_0 \\
w_1 &= \widetilde{r}_0/\gamma_0 \\
p_1^V &= v_1 \\
p_1^W &= w_1
\end{aligned}$$

for $k = 1, \ldots$

$$\begin{aligned}
\varphi_k &= \frac{(p_k^W)^T \mathbf{A} p_k^V}{w_k^T v_k} \\
\mathbf{t} &= \mathbf{A} p_k^V - \varphi_k v_k \\
\gamma_k &= \|\mathbf{t}\| \\
v_{k+1} &= \mathbf{t}/\gamma_k \\
\mathbf{t} &= \mathbf{A}^T p_k^W - \varphi_k w_k \\
w_{k+1} &= \mathbf{t}/\gamma_k \\
\psi_k &= \frac{\gamma_k}{\varphi_k} \frac{w_{k+1}^T v_{k+1}}{w_k^T v_k} \\
p_{k+1}^V &= v_{k+1} - \psi_k p_k^V \\
p_{k+1}^W &= w_{k+1} - \psi_k p_k^W
\end{aligned}$$

end for

1.3 Metody s minimálními a galerkinovskými kvazi-rezidui

Z předchozího odstavce víme, jak sestrojit Lanczosovu a Arnoldiho bázi Krylovova prostoru. Nyní stojíme před situací, kdy chceme řešit soustavu (1.1) a hledáme k-tou aproximaci řešení x_k ve varietě $x_0 + \mathcal{K}_k(\mathbf{A}, r_0)$ ve tvaru

$$x_k = x_0 + \mathbf{V}_k z_k, \tag{1.67}$$

kde $\mathbf{V}_k \in \mathbb{R}^{n\times k}$ je matice, jejíž sloupce postupně tvoří bázové vektory $v_1, \ldots, v_k$ splňující podmínku (1.9) a $z_k \in \mathbb{R}^k$ je vhodně zvolený vektor. Reziduový vektor r_k krylovovské metody příslušný k aproximaci x_k lze potom vyjádřit ve tvaru

$$\begin{aligned}
r_k &= b - \mathbf{A}x_k \\
&= b - \mathbf{A}(x_0 + \mathbf{V}_k z_k) \\
&= r_0 - \mathbf{A}\mathbf{V}_k z_k \\
&= \|r_0\| v_1 - \mathbf{V}_{k+1}\underline{\mathbf{H}}_k z_k \\
&= \mathbf{V}_{k+1}(\|r_0\| e_1^{(k+1)} - \underline{\mathbf{H}}_k z_k) = \mathbf{V}_{k+1}\, q_k,
\end{aligned} \tag{1.68}$$

kde $e_1^{(k+1)} \in \mathbb{R}^{k+1}$, $e_1^{(k+1)} \equiv (1, 0, \ldots, 0)^T$ a vektor $q_k \equiv \|r_0\| e_1^{(k+1)} - \underline{\mathbf{H}}_k z_k$, jež je souřadnicovým vektorem rezidua r_k v bázi dané vektory $v_1, \ldots, v_{k+1}$, je kvazi-reziduum uvažované v (1.12). Definujeme-li vektor z_k^M podmínkou

$$z_k^M \equiv \underset{z\in\mathbb{R}^k}{\arg\min}(\|\|r_0\| e_1^{(k+1)} - \underline{\mathbf{H}}_k z\|), \tag{1.69}$$

je minimální kvazi-reziduum q_k^M určeno vztahem

$$q_k^M = \|r_0\| e_1^{(k+1)} - \underline{\mathbf{H}}_k z_k^M.$$

Pro aproximace x_k^M a rezidua r_k^M krylovovské metody s minimálními kvazi-rezidui platí

$$x_k^M = x_0 + \mathbf{V}_k z_k^M, \qquad r_k^M = \mathbf{V}_{k+1}\, q_k^M.$$

Pokud při odvozování tvaru kvazi-rezidua (1.68) použijeme vztahu (1.16), dostáváme

$$\begin{aligned} r_k &= r_0 - \mathbf{A}\mathbf{V}_k z_k \\ &= \|r_0\| v_1 - \mathbf{V}_k \mathbf{H}_k z_k - h_{k+1k} v_{k+1} e_k^T z_k \\ &= \mathbf{V}_k(\|r_0\| e_1^{(k)} - \mathbf{H}_k z_k) - h_{k+1k} v_{k+1} (z_k)_k \end{aligned} \tag{1.70}$$

kde $(z_k)_k$ je poslední prvek vektoru z_k. Reziduum r_k je zřejmě násobkem posledního bázového vektoru v_{k+1} pokud platí

$$\|r_0\| e_1^{(k)} = \mathbf{H}_k z_k. \tag{1.71}$$

Definujeme-li vektor z_k^G jako řešení soustavy (1.71), je zřejmě vektor

$$q_k^G = \|r_0\| e_1^{(k+1)} - \underline{\mathbf{H}}_k z_k^G$$

galerkinovským kvazi-reziduem. Aproximace x_k^G a rezidua r_k^G krylovovské metody s galerkinovskými kvazi-rezidui lze vyjádřit ve tvaru

$$x_k^G = x_0 + \mathbf{V}_k z_k^G, \qquad r_k^G = \mathbf{V}_{k+1}\, q_k^G.$$

V následujících odstavcích se budeme zabývat algoritmickými postupy, pomocí nichž lze efektivně vypočíst vektory z_k^M, z_k^G, aproximace x_k^M, x_k^G a rezidua r_k^M, r_k^G krylovovských metod s minimálními a galerkinovskými kvazi-rezidui. Ukážeme, jaké vztahy platí mezi vektory obou typů metod (používajících stejnou bázi).

1.3.1 Krylovovská metoda s minimálními kvazi-rezidui

Věnujme se otázce konstrukce vektoru $z_k^M \in \mathbb{R}^k$. Uvažujme **QR**-rozklad matice $\underline{\mathbf{H}}_k$,

$$\underline{\mathbf{H}}_k = \mathbf{Q}_k \underline{\mathbf{R}}_k, \tag{1.72}$$

kde $\mathbf{Q}_k \in \mathbb{R}^{(k+1)\times(k+1)}$, $\mathbf{Q}_k^T\mathbf{Q}_k = \mathbf{I}_{k+1}$ ($\mathbf{I}_{k+1}$ je matice identity řádu $k+1$) a $\underline{\mathbf{R}}_k \in \mathbb{R}^{(k+1)\times k}$ je horní trojúhelníková matice, která má v posledním řádku samé nuly. Protože platí

$$\|\|r_0\| e_1^{(k+1)} - \underline{\mathbf{H}}_k z\,\| \;=\; \|\mathbf{Q}_k(\mathbf{Q}_k^T\|r_0\| e_1^{(k+1)} - \underline{\mathbf{R}}_k z)\| \;=\; \|\mathbf{Q}_k^T\|r_0\| e_1^{(k+1)} - \underline{\mathbf{R}}_k z\,\|,$$

je zřejmě

$$z_k^M = \underset{z\in\mathbb{R}^k}{\arg\min}(\|\mathbf{Q}_k^T\|r_0\| e_1^{(k+1)} - \underline{\mathbf{R}}_k z\,\|).$$

Označíme-li vektor $\mathbf{Q}_k^T\|r_0\| e_1^{(k+1)}$ symbolem $\underline{u}^{(k)} \in \mathbb{R}^{k+1}$, platí

$$\|\underline{u}^{(k)} - \underline{\mathbf{R}}_k z\,\|^2 = \|u^{(k)} - \mathbf{R}_k z\,\|^2 + |\eta_{k+1}|^2, \tag{1.73}$$

kde vektor $u^{(k)} \in \mathbb{R}^k$ vznikne z vektoru $\underline{u}^{(k)} \in \mathbb{R}^{k+1}$ vynecháním posledního prvku, matice $\mathbf{R}_k \in \mathbb{R}^{k\times k}$ vznikne z matice $\underline{\mathbf{R}}_k \in \mathbb{R}^{(k+1)\times k}$ vynecháním posledního řádku a η_{k+1} je poslední prvek vektoru $\underline{u}^{(k)}$. Z (1.73) plyne, že vektor z_k^M realizující minimum normy na levé straně (1.73) je řešením soustavy

$$\mathbf{R}_k z \;= u^{(k)}.$$

Povšimněme si dále, že

$$\|q_k^M\| = \|\underline{u}^{(k)} - \underline{\mathbf{R}}_k z_k^M\| = \sqrt{\|u^{(k)} - \mathbf{R}_k z_k^M\|^2 + |\eta_{k+1}|^2} = |\eta_{k+1}|. \tag{1.74}$$

Věnujme se realizaci **QR**-rozkladu matice $\underline{\mathbf{H}}_k$. Definujme matice Givensových rotací $\mathbf{G}_j^{(k)} \in \mathbb{R}^{(k+1)\times(k+1)}$ pro $j = 1, \ldots, k$ tak, že

$$\mathbf{G}_j^{(k)} = \begin{pmatrix} \mathbf{I}_{j-1} & & & \\ & c_j & -s_j & \\ & s_j & c_j & \\ & & & \mathbf{I}_{k-j} \end{pmatrix},$$

$s_j^2 + c_j^2 = 1$ a matice $\mathbf{G}_j^{(k)} \ldots \mathbf{G}_1^{(k)} \underline{\mathbf{H}}_k$ má prvek na místě $(j+1, j)$ roven nule. Položíme-li nyní $\mathbf{Q}_k^T \equiv \mathbf{G}_k^{(k)} \ldots \mathbf{G}_1^{(k)}$ potom platí $\mathbf{Q}_k^T \underline{\mathbf{H}}_k = \underline{\mathbf{R}}_k$, kde $\underline{\mathbf{R}}_k \in \mathbb{R}^{(k+1)\times k}$ je horní trojúhelníková matice s nulovým posledním řádkem a $\mathbf{Q}_k \underline{\mathbf{R}}_k$ je **QR**-rozklad matice $\underline{\mathbf{H}}_k$.

Protože v k-tém kroku vznikne matice $\underline{\mathbf{H}}_k$ z $\underline{\mathbf{H}}_{k-1}$ přidáním sloupce $(h_{1k}, \ldots, h_{k+1k})^T$, použijí se Givensovy rotace z $(k-1)$-ního kroku i v k-tém kroku, pouze se zvětší jejich řád. Zřejmě platí

$$\begin{aligned} \mathbf{Q}_k^T &= \mathbf{G}_k^{(k)} \ldots \mathbf{G}_1^{(k)} \\ &= \mathbf{G}_k^{(k)} \begin{pmatrix} \mathbf{G}_{k-1}^{(k-1)} & \mathbf{o} \\ \mathbf{o}^T & 1 \end{pmatrix} \ldots \begin{pmatrix} \mathbf{G}_1^{(k-1)} & \mathbf{o} \\ \mathbf{o}^T & 1 \end{pmatrix} = \mathbf{G}_k^{(k)} \begin{pmatrix} \mathbf{Q}_{k-1}^T & \mathbf{o} \\ \mathbf{o}^T & 1 \end{pmatrix}. \end{aligned} \tag{1.75}$$

Matici $\mathbf{G}_k^{(k)}$ určíme tak, že $c_k^2 + s_k^2 = 1$ a že nuluje prvek na místě $(k+1, k)$ matice

$$\begin{pmatrix} \mathbf{Q}_{k-1}^T & \\ & 1 \end{pmatrix} \underline{\mathbf{H}}_k. \tag{1.76}$$

Matici (1.76) můžeme vyjádřit ve tvaru

$$\begin{pmatrix} \mathbf{Q}_{k-1}^T & \\ & 1 \end{pmatrix} \underline{\mathbf{H}}_k = \begin{pmatrix} \mathbf{Q}_{k-1}^T & \\ & 1 \end{pmatrix} \begin{pmatrix} \underline{\mathbf{H}}_{k-1} & \begin{pmatrix} h_{k1} \\ \vdots \\ h_{kk} \end{pmatrix} \\ & h_{k+1k} \end{pmatrix} = \begin{pmatrix} \underline{\mathbf{R}}_{k-1} & \mathbf{Q}_{k-1}^T \begin{pmatrix} h_{k1} \\ \vdots \\ h_{kk} \end{pmatrix} \\ 0 \ldots 0 & h_{k+1k} \end{pmatrix}.$$

Označíme-li

$$t^{(k)} \equiv \mathbf{Q}_{k-1}^T \begin{pmatrix} h_{k1} \\ \vdots \\ h_{kk} \end{pmatrix} = \begin{pmatrix} t_1^{(k)} \\ \vdots \\ t_k^{(k)} \end{pmatrix}, \tag{1.77}$$

plyne z požadavku nulování prvku na místě $(k+1, k)$ matice (1.76), že

$$c_k = \frac{t_k^{(k)}}{\sqrt{(t_k^{(k)})^2 + h_{k+1k}^2}} \quad \text{a} \quad s_k = -\frac{h_{k+1,k}}{\sqrt{(t_k^{(k)})^2 + h_{k+1k}^2}}. \tag{1.78}$$

Pro matici $\underline{\mathbf{R}}_k$ potom zřejmě platí

$$\underline{\mathbf{R}}_k = \begin{pmatrix} \mathbf{I}_{k-1} & & \\ & c_k & -s_k \\ & s_k & c_k \end{pmatrix} \begin{pmatrix} \underline{\mathbf{R}}_{k-1} & t^{(k)} \\ \mathbf{o}^T & h_{k+1k} \end{pmatrix} = \begin{pmatrix} & & t_1^{(k)} \\ & \mathbf{R}_{k-1} & \vdots \\ & & t_{k-1}^{(k)} \\ & & r_{kk} \\ & & 0 \end{pmatrix}, \tag{1.79}$$

kde prvek r_{kk} na místě (k,k) matice $\mathbf{R}_k$ vypočteme jako

$$\begin{aligned} r_{kk} &= c_k t_k^{(k)} - s_k h_{k+1k} \\ &= \frac{(t_k^{(k)})^2}{\sqrt{(t_k^{(k)})^2 + h_{k+1k}^2}} + \frac{h_{k+1,k}^2}{\sqrt{(t_k^{(k)})^2 + h_{k+1k}^2}} \\ &= \sqrt{(t_k^{(k)})^2 + h_{k+1k}^2}. \end{aligned} \tag{1.80}$$

Matice $\mathbf{R}_k$ má zřejmě na diagonále nenulové prvky. Podle (1.75) je potom

$$\underline{u}^{(k)} = \begin{pmatrix} \mathbf{I}_{k-1} & & \\ & c_k & -s_k \\ & s_k & c_k \end{pmatrix} \begin{pmatrix} \underline{u}^{(k-1)} \\ 0 \end{pmatrix} = \begin{pmatrix} u^{(k-1)} \\ c_k \eta_k \\ s_k \eta_k \end{pmatrix} \tag{1.81}$$

a indukcí dostáváme (za použití $\eta_1 = \|r_0\|$)

$$\eta_{k+1} = s_k \eta_k = s_k s_{k-1} \dots s_1 \|r_0\|. \tag{1.82}$$

Formule (1.77), (1.78), (1.79) a (1.81) nám umožňují počítat matici $\mathbf{R}_k$ a vektor $u^{(k)}$ pomocí hodnot dostupných z předchozí iterace. Ze soustavy $\mathbf{R}_k z = u^{(k)}$ lze poté vypočíst vektor z_k^M a vyjádřit k-té přiblížení x_k^M.

V následujícím lemmatu uvádíme návod, jak lze vypočítat reziduum r_k^M na základě znalosti rezidua r_{k-1}^M a posledního vektoru báze v_{k+1}. Lemma bude především důležité pro určení vztahu mezi metodami s minimálními a galerkinovskými kvazi-rezidui, které používají stejnou bázi, a při případném odvozování algoritmů krylovovských metod.

Lemma 1.1 *Pro reziduum r_k^M krylovovské metody s minimálním kvazi-reziduem platí*

$$r_k^M = r_{k-1}^M s_k^2 + v_{k+1} c_k \eta_{k+1}. \tag{1.83}$$

Důkaz. Vektor r_k^M můžeme vyjádřit ve tvaru

$$\begin{aligned} r_k^M &= \mathbf{V}_{k+1}\left(\|r_0\| e_1^{(k+1)} - \underline{\mathbf{H}}_k z_k^M\right) = \mathbf{V}_{k+1}\left(\|r_0\| e_1^{(k+1)} - \mathbf{Q}_k \begin{pmatrix} \mathbf{R}_k \\ \mathbf{o} \end{pmatrix} \mathbf{R}_k^{-1} u^{(k)}\right) \\ &= \mathbf{V}_{k+1}\left(\|r_0\| e_1^{(k+1)} - \mathbf{Q}_k \begin{pmatrix} u^{(k)} \\ 0 \end{pmatrix}\right) = \mathbf{V}_{k+1}\mathbf{Q}_k\left(\mathbf{Q}_k^T \|r_0\| e_1^{(k+1)} - \begin{pmatrix} u^{(k)} \\ 0 \end{pmatrix}\right) \\ &= \mathbf{V}_{k+1}\mathbf{Q}_k\left(\begin{pmatrix} u^{(k)} \\ \eta_{k+1} \end{pmatrix} - \begin{pmatrix} u^{(k)} \\ 0 \end{pmatrix}\right) = \mathbf{V}_{k+1}\mathbf{Q}_k \begin{pmatrix} \mathbf{o} \\ \eta_{k+1} \end{pmatrix}. \end{aligned} \tag{1.84}$$

Uvažujeme-li matici $\mathbf{Q}_k$ ve tvaru (1.75), platí

$$\begin{aligned} r_k^M &= (\mathbf{V}_k, v_{k+1}) \begin{pmatrix} \mathbf{Q}_{k-1} & \mathbf{o} \\ \mathbf{o} & 1 \end{pmatrix} \mathbf{G}_k^T \begin{pmatrix} \mathbf{o} \\ \eta_{k+1} \end{pmatrix} = \eta_{k+1} (\mathbf{V}_k \mathbf{Q}_{k-1}, v_{k+1}) \begin{pmatrix} \mathbf{o} \\ s_k \\ c_k \end{pmatrix} = \\ &= s_k \eta_{k+1} \mathbf{V}_k \mathbf{Q}_{k-1} \begin{pmatrix} \mathbf{o} \\ 1 \end{pmatrix} + \eta_{k+1} c_k v_{k+1}. \end{aligned} \tag{1.85}$$

Ze vztahů

$$r_k^M = s_k \eta_{k+1} \mathbf{V}_k \mathbf{Q}_{k-1} \begin{pmatrix} \mathbf{o} \\ 1 \end{pmatrix} + \eta_{k+1} c_k v_{k+1} \quad \text{a} \quad r_{k-1}^M = \mathbf{V}_k \mathbf{Q}_{k-1} \begin{pmatrix} \mathbf{o} \\ \eta_k \end{pmatrix}$$

plyne (s využitím (1.85) a (1.82)) vyjádření

$$r_k^M = s_k \frac{\eta_{k+1}}{\eta_k} r_{k-1}^M + \eta_{k+1} c_k v_{k+1} = s_k^2 r_{k-1}^M + v_{k+1} c_k \eta_{k+1}. \qquad \square$$

1.3.2 Krylovovská metoda s galerkinovskými kvazi-rezidui

Úkolem tohoto odstavce je určit vektor z_k^G, jež je potřebný k vyjádření aproximace x_k^G a ukázat způsob, jak lze na základě znalosti posledního bázového vektoru v_{k+1} vypočíst reziduový vektor r_k^G.

Vektor z_k^G je řešením soustavy (1.71). Řešme tuto soustavu pomocí **QR**-rozkladu matice $\mathbf{H}_k$. Pro obnovování **QR**-rozkladů matic $\mathbf{H}_k$ můžeme využít poznatků z předchozího odstavce. Z (1.72), (1.75) a (1.76) plyne

$$\underline{\mathbf{H}}_k = \mathbf{Q}_k \underline{\mathbf{R}}_k = \begin{pmatrix} \mathbf{Q}_{k-1} & \\ & 1 \end{pmatrix} \begin{pmatrix} \mathbf{I}_{k-1} & & \\ & c_k & s_k \\ & -s_k & c_k \end{pmatrix} \underline{\mathbf{R}}_k. \tag{1.86}$$

Neuvažujeme-li poslední řádky matic na pravé a levé straně (1.86), dostáváme

$$\mathbf{H}_k = \mathbf{Q}_{k-1} \begin{pmatrix} \mathbf{I}_{k-1} & \\ & c_k \end{pmatrix} \mathbf{R}_k. \tag{1.87}$$

a (1.87) je zřejmě **QR**-rozklad matice $\mathbf{H}_k$. Dosadíme-li matici $\mathbf{H}_k$ ve tvaru (1.87) do soustavy (1.71), je zřejmě vektor z_k^G řešením soustavy

$$\mathbf{R}_k z = \begin{pmatrix} \mathbf{I}_{k-1} & \\ & c_k^{-1} \end{pmatrix} \mathbf{Q}_{k-1}^T \|r_0\| e_1^{(k)}. \tag{1.88}$$

Použijeme-li značení z (1.73), t.j. $\underline{u}^{(k-1)} = \mathbf{Q}_{k-1}^T \|r_0\| e_1^{(k)}$, lze vektor pravé strany soustavy (1.88) psát ve tvaru

$$\begin{pmatrix} \mathbf{I}_{k-1} & \\ & c_k^{-1} \end{pmatrix} \underline{u}^{(k-1)} = \begin{pmatrix} \mathbf{I}_{k-1} & \\ & c_k^{-1} \end{pmatrix} \begin{pmatrix} u^{(k-1)} \\ \eta_k \end{pmatrix} = \begin{pmatrix} u^{(k-1)} \\ \eta_k c_k^{-1} \end{pmatrix}$$

a z_k^G je řešením soustavy

$$\mathbf{R}_k z = \begin{pmatrix} u^{(k-1)} \\ \eta_k c_k^{-1} \end{pmatrix}. \tag{1.89}$$

Následující lemma vypovídá o tom, jak vypočíst reziduum r_k^G na základě znalosti posledního bázového vektoru a hodnot z **QR** rozkladu matice $\mathbf{H}_k$.

Lemma 1.2 *Pro reziduum r_k^G krylovovské metody s galerkinovskými kvazi-rezidui platí*

$$r_k^G = v_{k+1} \frac{\eta_{k+1}}{c_k}. \tag{1.90}$$

Důkaz. Podle (1.70) je $r_k^G = -v_{k+1} h_{k+1k} (z_k^G)_k$, kde $(z_k^G)_k$ je poslední prvek vektoru z_k^G. Zbývá vyjádřit prvek $(z_k^G)_k$. Poslední prvek z_k^G je zřejmě dle (1.88) roven $\eta_k/(r_{kk} c_k)$, přičemž r_{kk} je prvek na místě (k,k) matice $\mathbf{R}_k$, který vzhledem k rovnosti (1.86) splňuje rovnici $h_{k+1k} = -s_k r_{kk}$. Platí

$$r_k^G = -v_{k+1} h_{k+1k} \frac{\eta_k}{r_{kk} c_k} = v_{k+1} \frac{s_k \eta_k}{c_k} = v_{k+1} \frac{\eta_{k+1}}{c_k}. \qquad \square$$

1.3.3 Vztahy mezi metodami

Použití **QR**-rozkladu matice $\underline{\mathbf{H}}_k$ a $\mathbf{H}_k$ k řešení problémů (1.69) a (1.71) nám umožní odkrýt souvislost mezi krylovovskými metodami s minimálními a galerkinovskými kvazi-rezidui.

Lemma 1.3 *Mezi vektory určenými krylovovskými metodami s minimálními a galerkinovskými kvazi-rezidui, jež používají stejnou bázi, platí vztahy*

$$r_k^M = s_k^2 r_{k-1}^M + c_k^2 r_k^G, \qquad x_k^M = s_k^2 x_{k-1}^M + c_k^2 x_k^G, \tag{1.91}$$

$$q_k^M = \begin{pmatrix} q_{k-1}^M \\ 0 \end{pmatrix} s_k^2 + c_k^2 q_k^G, \qquad z_k^M = \begin{pmatrix} z_{k-1}^M \\ 0 \end{pmatrix} s_k^2 + c_k^2 z_k^G. \tag{1.92}$$

Hodnotu $|\eta_{k+1}|$ je možné počítat rekurentně ze vzorce

$$\frac{1}{\eta_{k+1}^2} = \frac{1}{\eta_k^2} + \frac{1}{\|r_k^G\|^2}. \tag{1.93}$$

Navíc platí

$$\|r_k^G\| = \frac{\|q_k^M\|}{\sqrt{1 - \frac{\|q_k^M\|^2}{\|q_{k-1}^M\|^2}}}, \qquad \|r_k^G\| = \frac{|\eta_{k+1}|}{|c_k|}. \tag{1.94}$$

Důkaz. Dosadíme-li do (1.83) za vektor v_{k+1} podle (1.90) vektor $c_k/\eta_{k+1} r_k^G$ a využijeme-li vztahu mezi reziduem a aproximací, dostáváme vztahy (1.91).

Protože platí $s_k^2 + c_k^2 = 1$ a $\eta_{k+1} = s_k \eta_k$ je podle (1.90)

$$\frac{1}{\eta_{k+1}^2} = \frac{s_k^2 + c_k^2}{s_k^2 \eta_k^2} = \frac{1}{\eta_k^2} + \frac{c_k^2}{\eta_{k+1}^2} = \frac{1}{\eta_k^2} + \frac{1}{\|r_k^G\|^2}. \tag{1.95}$$

Platí tedy vztah (1.93). Uvědomíme-li si, že $|\eta_{k+1}| = \|q_k^M\|$, dostáváme vyjádřením $\|r_k^G\|$ ze vztahu (1.93) první část (1.94). Druhá plyne z (1.90).

Ze vztahů (1.91), (1.67), z definice kvazi-rezidua a vztahu $c_k^2 + s_k^2 = 1$ plyne

$$\mathbf{V}_k z_k^M = s_k^2 \mathbf{V}_{k-1} z_{k-1}^M + c_k^2 \mathbf{V}_k z_k^G = s_k^2 \mathbf{V}_k \begin{pmatrix} z_{k-1}^M \\ 0 \end{pmatrix} + c_k^2 \mathbf{V}_k z_k^G,$$

$$\mathbf{V}_{k+1} q_k^M = s_k^2 \mathbf{V}_k q_{k-1}^M + c_k^2 \mathbf{V}_{k+1} q_k^G = s_k^2 \mathbf{V}_{k+1} \begin{pmatrix} q_{k-1}^M \\ 0 \end{pmatrix} + c_k^2 \mathbf{V}_{k+1} q_k^G.$$

Z právě uvedených vztahů a z lineární nezávislosti vektorů v_i již plyne (1.92). □

1.4 Algoritmy krylovovských metod

Z předchozích dvou sekcí máme již dostatek prostředků na formulování algoritmů uvažovaných krylovovských metod GMRES, FOM, LM a QMR. Je zřejmé, že jednotlivé metody mohou být reprezentovány různým způsobem v závislosti na volbě algoritmů realizující jednotlivé fáze výpočtu. V následujících odstavcích provedeme odvození některých algoritmů uvažovaných krylovovských metod.

Algoritmus 1.4. GMRES

input $x_0,\ \mathbf{A},\ b$

initialization

$$\begin{aligned} r_0 &= b - \mathbf{A}x_0 \\ &\eta_1 = \|r_0\| \\ v_1 &= r_0/\eta_1 \end{aligned}$$

for $k = 1,\ \ldots$

$$\begin{aligned} w &= \mathbf{A}v_k \\ &\delta = h_{1k} = v_1^T w \\ w &= w - h_{1k}v_1 \end{aligned}$$

for $i = 2,\ \ldots,\ k$

$$\begin{aligned} &h_{ik} = v_i^T w \\ w &= w - h_{ik}v_i \\ &r_{i-1,k} = c_{i-1}\delta - s_{i-1}h_{ik} \\ &\delta = s_{i-1}\delta + c_{i-1}h_{ik} \end{aligned}$$

end for

$$\begin{aligned} &h_{k+1k} = \|w\| \\ v_{k+1} &= w/h_{k+1k} \\ &r_{kk} = (\delta^2 + h_{k+1k}^2)^{1/2} \\ &c_k = \delta/r_{kk} \\ &s_k = -h_{k+1k}/r_{kk} \\ &u_k = c_k\eta_k \\ &\eta_{k+1} = s_k\eta_k \end{aligned}$$

end for

compute z_k^M
$x_k^M = x_0 + \mathbf{V}_k z_k^M$

1.4.1 Algoritmus GMRES

Pro naši implementaci metody GMRES budeme volit za algoritmus počítající Arnoldiho bázi modifikovaný Arnoldiho algoritmus 1.1 a pro konstrukci vektoru z_k^M definovaného v (1.69) použijeme techniku **QR**-rozkladu z předchozí sekce.

Pro normu rezidua r_k^M metody GMRES platí

$$\|r_k^M\| = \sqrt{q_k^{M\,T}\mathbf{V}_{k+1}^T\mathbf{V}_{k+1}q_k^M} = \|q_k^M\| \tag{1.96}$$

a informaci o velikosti normy rezidua jsme zřejmě podle (1.74) schopni zjistit z $|\eta_{k+1}|$. Abychom mohli vypočítat vektor z_k^M a tedy i x_k^M, je nutné v každém kroku znát matici $\mathbf{R}_k$ a vektor $u^{(k)}$. Popišme postup počítání této matice a vektoru v k-tém kroku.

Předpokládejme, že máme k dispozici matici $\mathbf{R}_{k-1}$, vektor $u^{(k-1)}$, matici $\mathbf{V}_k$ a hodnoty $s_1, \ldots, s_{k-1}$, $c_1, \ldots, c_{k-1}$. Pomocí modifikovaného Arnoldiho algoritmu vypočteme vektor v_{k+1}, jež nám společně s maticí $\mathbf{V}_k$ definuje matici $\mathbf{V}_{k+1}$. Zároveň získáme z algoritmu poslední sloupec matice $\underline{\mathbf{H}}_k$ daný koeficienty $h_{k1}, \ldots, h_{k+1k}$. K vypočtení hodnot c_k a s_k potřebujeme znát kromě hodnoty h_{k+1k} ještě poslední prvek vektoru $t^{(k)}$. Tento prvek podle (1.77) vypočteme vynásobením vektoru koeficientů $h_{k1}, \ldots, h_{kk}$ příslušnými Givensovými rotacemi definovanými hodnotami $s_1, \ldots, s_{k-1}$, $c_1, \ldots, c_{k-1}$. Nyní nám již nic nebrání ve vypočtení hodnot c_k a s_k podle (1.78). Dále dle (1.79), (1.80) a (1.81) vypočteme matici $\mathbf{R}_k$, vektor $u^{(k)}$ a číslo η_{k+1}, jehož absolutní hodnota udává normu příslušného rezidua, na jejímž základě se můžeme rozhodnout, zda ze soustavy s horní trojúhelníkovou maticí vypočíst vektor z_k^M a použít ho k vyjádření x_k^M nebo zda pokračovat v dalším

kroku GMRES. Poznamenejme ještě, že vektor $t^{(k)}$ lze počítat již při běhu modifikovaného Arnoldiho algoritmu 1.1. K samotnému algoritmu pouze dodejme, že symbolem r_{ij} značíme prvky horní trojúhelníkové matice $\mathbf{R}_k$. V následujícím kroku vždy pouze dopočteme další sloupec této matice. Pomocné číslo δ je potřebné při násobení Givensových rotací posledním sloupcem matice $\underline{\mathbf{H}}_k$ a symbol u_k značí vždy k-tý prvek vektoru pravé strany $u^{(k)}$. Ten se při další iteraci změní ve vektor $u^{(k+1)}$ přidáním poslední složky podle (1.81).

1.4.2 Algoritmus FOM

Volme opět za algoritmus počítající Arnoldiho bázi modifikovaný Arnoldiho algoritmus 1.1 a počítejme vektor z_k^G užitím **QR**-rozkladu matice $\mathbf{H}_k$ z (1.89). Algoritmus metody FOM neuvádíme, protože je v tomto případě shodný s algoritmem GMRES 1.4 až na výsledné vypočtení aproximace řešení x_k^G podle (1.89) a (1.67). Informaci o velikosti normy rezidua r_k^G lze vyčíst z (1.94).

Reziduum r_k^G metody FOM je dáno průnikem přímky určené vektorem Arnoldiho báze v_{k+1} s varietou $r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$. Může nastat situace, kdy tento průnik neexistuje a potom nelze reziduum požadovaných vlastností sestrojit. V algoritmu FOM se projeví tato situace tak, že platí $c_k = 0$ a matice $\mathbf{H}_k$ je singulární. Lze však pokračovat v konstrukci ortonormální báze prostoru a počítat pouze taková rezidua a aproximace, jejichž existence je zaručena.

Poznamenejme, že podle (1.94) a (1.96) platí zřejmě mezi normami reziduí metody GMRES a FOM vztah

$$\|r_k^G\| = \frac{\|r_k^M\|}{\sqrt{1 - \frac{\|r_k^M\|^2}{\|r_{k-1}^M\|^2}}}. \tag{1.97}$$

Klesá-li norma rezidua GMRES rychle, plyne ze vztahu (1.97)

$$\|r_k^G\| \approx \|r_k^M\|$$

a norma rezidua GMRES je blízká normě rezidua FOM. Stagnuje-li konvergence GMRES, t.j. pozorujeme-li na konvergenční křivce plošinu, lze podle (1.97) očekávat nárůst normy rezidua metody FOM. V praxi je plošina na reziduové křivce GMRES svázána s převýšením (peak) u metody FOM.

1.4.3 Algoritmus QMR

V tomto odstavci odvodíme jeden z možných algoritmů metody QMR [14]. Využijeme přitom algoritmu 1.2 (NL), kterým budeme počítat vektory Lanczosovy báze. Soustavu v (1.69) budeme poté řešit užitím **QR**-rozkladu. Protože by klasický postup počítání aproximace x_k^M podle (1.67) vyžadoval uchovávání Lanczosových vektorů v paměti počítače, je nutná modifikace, využívající struktury matice $\mathbf{R}_k$.

Nechť jsou tedy sloupce matice $\mathbf{V}_k$ tvořeny Lanczosovými vektory, $\mathbf{T}_k \in \mathbb{R}^{k\times k}$ nechť je třídiagonální matice splňující (1.40) (matici $\mathbf{H}_k$ jsme v případě NL pouze označili $\mathbf{T}_k$). Uvažujme **QR**-rozklad matice $\mathbf{T}_k$ podle (1.87). Je zřejmé, že $\mathbf{R}_k$, jež vznikla aplikací matic Givensových rotací na $\mathbf{T}_k$, je regulární, horní trojúhelníková matice, která má nenulovou pouze diagonálu a dvě horní naddiagonály. Definujme matici

$$\mathbf{M}_k \equiv \mathbf{V}_k \mathbf{R}_k^{-1}. \tag{1.98}$$

Sloupce $m_1, \ldots, m_k$ matice $\mathbf{M}_k$ tvoří podle (1.98) bázi prostoru $\mathcal{K}_k(\mathbf{A}, r_0)$ a lze je počítat na základě (1.98) podle vztahů

$$m_1 \;=\; r_{11}^{-1} v_1,$$

$$
\begin{aligned}
m_2 &= r_{22}^{-1}(v_2 - r_{12}m_1),\\
m_k &= r_{kk}^{-1}(v_k - r_{k-1k}m_{k-1} - r_{k-2k}m_{k-2})
\end{aligned}
\tag{1.99}
$$

pro $k > 2$. Odečtením $x_{k-1}^M = x_0 + \mathbf{M}_{k-1}u^{(k-1)}$ od $x_k^M = x_0 + \mathbf{M}_k u^{(k)}$ a využitím vztahu mezi $u^{(k-1)}$ a $u^{(k)}$ (1.81) dostáváme

$$
x_k^M = x_{k-1}^M + \mathbf{M}_k \begin{pmatrix} u^{(k-1)} \\ c_k\eta_k \end{pmatrix} - \mathbf{M}_{k-1}u^{(k-1)} = x_{k-1}^M + c_k\eta_k m_k. \tag{1.100}
$$

Zbývá odvodit rekurentní vztahy pro počítání prvků matice $\mathbf{R}_k$, nezbytných k vypočtení vektorů m_k, a dále vztahy pro hodnoty s_k, c_k a η_k. Uvážíme-li označení (1.32)–(1.33), dále vztahy (1.78)–(1.82) a definujeme-li $\gamma_0 = 0$, platí $\eta_1 = \|r_0\|$ a vztahy mají tvar

$\mathbf{k = 1}$			$\mathbf{k = 2}$			$\mathbf{k > 2}$		
r_{11}	$=$	$(\alpha_1^2 + \gamma_1^2)^{1/2}$,	r_{12}	$=$	$c_1\beta_1 - s_1\alpha_2$,	r_{k-2k}	$=$	$-s_{k-2}\beta_{k-1}$,
c_1	$=$	α_1/r_{11},	δ	$=$	$s_1\beta_1 + c_1\alpha_2$,	δ	$=$	$c_{k-2}\beta_{k-1}$,
s_1	$=$	$-\gamma_1/r_{11}$,	r_{22}	$=$	$(\delta^2 + \gamma_1^2)^{1/2}$,	r_{k-1k}	$=$	$c_{k-1}\delta - s_{k-1}\alpha_k$,
η_2	$=$	$s_1\eta_1$,	c_2	$=$	δ/r_{22},	δ	$=$	$s_{k-1}\delta + c_{k-1}\alpha_k$,
			s_2	$=$	$-\gamma_2/r_{22}$,	r_{kk}	$=$	$(\delta^2 + \gamma_k^2)^{1/2}$,
			η_3	$=$	$s_2\eta_2$,	c_k	$=$	δ/r_{kk},
						s_k	$=$	$-\gamma_k/r_{kk}$,
						η_{k+1}	$=$	$s_k\eta_k$.

Dodejme, že k počítání reziduového vektoru užijeme vztah (1.83). Nyní je již možné zformulovat algoritmus 1.5 (QMR). Podobně by šel formulovat algoritmus metody QMR založený na algoritmu 1.3 (BiCV). V tomto případě bychom využili vztahů (1.65), (1.66) mezi koeficienty α_k, β_{k-1} a φ_k, ψ_{k-1} a hodnoty s_k, c_k a η_k bychom počítali přímo z koeficientů φ_k, ψ_{k-1}. Rezidua a aproximace metody QMR je též možno vypočíst za použití vztahů (1.91) a algoritmu vytvářejícího vektory x_k^G a r_k^G. V tomto případě potřebujeme znát k vypočtení x_k^M a r_k^M ještě čísla c_k a s_k. Hodnoty c_k a s_k však lze lehce vyjádřit pomocí (1.93) a druhého vztahu v (1.94).

1.4.4 Algoritmus LM (BiCG)

Rezidua a aproximace Lanczosovy metody pro řešení systému lineárních rovnic jsou dány podmínkou (1.8) a k určení rezidua r_k^G lze využít vhodného násobku vektoru v_{k+1} Lanczosovy báze. Rekurenci pro počítání aproximace x_k^G potom odvodíme z rekurence pro reziduový vektor a vztahu $r_k^G = b - \mathbf{A}x_k^G$.

Uvažujme algoritmus 1.3 (BiCV). V algoritmu 1.3 jsme volili koeficient γ_k tak, aby byla norma vektoru v_{k+1} rovna jedné. Pokud však volíme

$$
\gamma_0 \equiv 1, \qquad\qquad \gamma_k \equiv -\varphi_k \tag{1.101}
$$

pro $k > 0$, potom lze indukcí ukázat, že platí $v_{k+1} \in r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$ a v_{k+1} je nutně reziduový vektor, t.j. $v_{k+1} = r_k^G$. Podobně označme vektor w_{k+1} s volbou koeficientů γ_k podle (1.101) jako $\widetilde{r}_k$. Definujeme-li pro $k \geq 0$ koeficienty $\boldsymbol{\alpha}_k$ a $\boldsymbol{\beta}_k$ vztahy

$$
\boldsymbol{\alpha}_k = \varphi_{k+1}^{-1}, \qquad\qquad \boldsymbol{\beta}_k \equiv -\psi_{k+1}, \tag{1.102}
$$

Algoritmus 1.5. QMR

input $\mathbf{A}$, b, x_0, $\widetilde{r}_0$

initialization

$$
\begin{aligned}
m_{-1} &= \mathbf{o} \\
&s_{-1} = 0 \\
&c_{-1} = 0 \\
v_0 &= \mathbf{o} \\
w_0 &= \mathbf{o} \\
m_0 &= \mathbf{o} \\
&\beta_0 = 0 \\
&\gamma_0 = 0 \\
&c_0 = 1 \\
&s_0 = 0 \\
r_0 &= b - \mathbf{A}x_0 \\
v_1 &= r_0/\|r_0\| \\
w_1 &= \widetilde{r}_0/\widetilde{r}_0^T v_1 \\
&\eta_1 = 1
\end{aligned}
$$

for $k = 1, \ldots$

$$
\begin{aligned}
&\alpha_k = w_k^T \mathbf{A} v_k \\
\mathbf{t} &= \mathbf{A}v_k - \alpha_k v_k - \beta_{k-1} v_{k-1} \\
&\gamma_k = \|\mathbf{t}\| \\
v_{k+1} &= \mathbf{t}/\gamma_k \\
\mathbf{t} &= \mathbf{A}^T w_k - \alpha_k w_k - \gamma_{k-1} w_{k-1} \\
&\beta_k = v_{k+1}^T \mathbf{t} \\
w_{k+1} &= \mathbf{t}/\beta_k \\
&r_{k-2k} = -s_{k-2}\beta_{k-1} \\
&\delta = c_{k-2}\beta_{k-1} \\
&r_{k-1k} = c_{k-1}\delta - s_{k-1}\alpha_k \\
&\delta = s_{k-1}\delta + c_{k-1}\alpha_k \\
&r_{kk} = (\delta^2 + \gamma_k^2)^{1/2} \\
&c_k = \delta/r_{kk} \\
&s_k = -\gamma_k/r_{kk} \\
&\eta_{k+1} = s_k \eta_k \\
m_k &= r_{kk}^{-1}\big(v_k - r_{k-1k} m_{k-1} - r_{k-2k} m_{k-2}\big) \\
x_k &= x_{k-1} + c_k \eta_k m_k \\
r_k &= s_k^2 r_{k-1} + c_k \eta_{k+1} v_{k+1}
\end{aligned}
$$

end for

a vektory p_k a $\widetilde{p}_k$ podle

$$
p_k \equiv p_{k+1}^V, \qquad \widetilde{p}_k \equiv p_{k+1}^W, \tag{1.103}
$$

lze rekurence

$$
\begin{aligned}
v_{k+2} &= \gamma_{k+1}^{-1}(\mathbf{A}p_{k+1}^V - \varphi_{k+1} v_{k+1}), & w_{k+2} &= \gamma_{k+1}^{-1}(\mathbf{A}^T p_{k+1}^W - \varphi_{k+1} w_{k+1}) \\
p_{k+2}^V &= v_{k+2} - \psi_{k+1} p_{k+1}^V, & p_{k+2}^W &= w_{k+2} - \psi_{k+1} p_{k+1}^W
\end{aligned}
$$

psát ve tvaru

$$
\begin{aligned}
r_{k+1}^G &= r_k^G - \boldsymbol{\alpha}_k \mathbf{A} p_k, & \widetilde{r}_{k+1}^G &= \widetilde{r}_k^G - \boldsymbol{\alpha}_k^T \mathbf{A} \widetilde{p}_k \\
p_{k+1} &= r_{k+1}^G + \boldsymbol{\beta}_k p_k, & \widetilde{p}_{k+1} &= \widetilde{r}_{k+1} + \boldsymbol{\beta}_k \widetilde{p}_k.
\end{aligned}
$$

Rozepsáním reziduí r_k^G podle $r_k^G = b - \mathbf{A}x_k^G$ a aplikací matice $\mathbf{A}^{-1}$ na pravou i levou stranu výše uvedené rekurence pro reziduum r_{k+1}^G získáme rekurenci pro x_{k+1}^G

$$
x_{k+1}^G = x_k^G + \boldsymbol{\alpha}_k p_k.
$$

Výsledný algoritmus 1.6 Lanczosovy metody nazveme algoritmus bikonjugovaných gradientů (BiCG) (v algoritmu zapisujeme rezidua a aproximace bez horního indexu G). Odvození BiCG lze provést odlišným způsobem než je uveden v této práci a sice přímo

Algoritmus 1.6. BiCG

input $x_0,\ \mathbf{A},\ b,\ \widetilde{r}_0$

initialization

$$\begin{aligned} r_0 &= b - \mathbf{A}x_0 \\ p_0 &= r_0 \\ \widetilde{p}_0 &= \widetilde{r}_0 \end{aligned}$$

for $k = 0,\ \ldots$

$$\begin{aligned} \boldsymbol{\alpha}_k &= \frac{\widetilde{r}_k^T r_k}{\widetilde{p}_k^T \mathbf{A} p_k} \\ x_{k+1} &= x_k + \boldsymbol{\alpha}_k p_k \\ r_{k+1} &= r_k - \boldsymbol{\alpha}_k \mathbf{A} p_k \\ \widetilde{r}_{k+1} &= \widetilde{r}_k - \boldsymbol{\alpha}_k \mathbf{A}^T \widetilde{p}_k \\ \boldsymbol{\beta}_k &= \frac{\widetilde{r}_{k+1}^T r_{k+1}}{\widetilde{r}_k^T r_k} \\ p_{k+1} &= r_{k+1} + \boldsymbol{\beta}_k p_k \\ \widetilde{p}_{k+1} &= \widetilde{r}_{k+1} + \boldsymbol{\beta}_k \widetilde{p}_k \end{aligned}$$

end for

z podmínek (1.8), jak jsme ukázali v [63]. Poznamenejme, že pomocí algoritmu BiCG lze zároveň konstruovat aproximace řešení soustavy

$$\mathbf{A}^T \widetilde{x} = \widetilde{b}. \tag{1.104}$$

Stačí položit $\widetilde{r}_0 = \widetilde{b} - \mathbf{A}^T \widetilde{x}_0$, kde $\widetilde{x}_0$ je počáteční aproximace řešení soustavy (1.104), a aproximace řešení soustavy (1.104) počítat rekurencí

$$\widetilde{x}_{k+1} = \widetilde{x}_k + \boldsymbol{\alpha}_k \widetilde{p}_k. \tag{1.105}$$

Vektory $\widetilde{r}_k$ zřejmě hrají roli reziduí, $\widetilde{r}_k = \widetilde{b} - \mathbf{A}^T \widetilde{x}_k$.

Algoritmus 1.6 lze lehce modifikovat tak, aby kromě reziduí a aproximací Lanczosovy metody počítal i rezidua a aproximace metody QMR. Do cyklu algoritmu BiCG stačí přidat rekurence (1.91) a hodnoty c_k a s_k počítat ze vzorců $c_k^2 + s_k^2 = 1$, (1.93) a (1.94).

1.5 Metody Lanczosova typu

Algoritmus BiCG počítá posloupnosti reziduí r_k, aproximací řešení x_k a vektorů p_k rekurencemi

$$r_{k+1} = r_k - \boldsymbol{\alpha}_k \mathbf{A} p_k, \tag{1.106}$$

$$x_{k+1} = x_k + \boldsymbol{\alpha}_k p_k, \tag{1.107}$$

$$p_{k+1} = r_{k+1} + \boldsymbol{\beta}_k p_k, \tag{1.108}$$

kde $p_0 \equiv r_0$, odvozených v předchozím odstavci. Aby bylo možné vypočíst koeficienty $\boldsymbol{\alpha}_k$ a $\boldsymbol{\beta}_k$, počítá BiCG navíc pomocné vektory $\widetilde{r}_k$ a $\widetilde{p}_k$, jichž může být použito k řešení duální soustavy (1.104). Pokud však je naším cílem pouze nalezení aproximace řešení soustavy (1.1), není plně využita informace, kterou nám poskytují pomocné vektory $\widetilde{r}_k$ a $\widetilde{p}_k$ vypočítávané za pomoci násobení maticí $\mathbf{A}^T$.

Ke zlepšení výkonnosti algoritmu přispěje následující úvaha. Z algoritmu BiCG plyne, že vektory r_k, p_k, $\widetilde{r}_k$ a $\widetilde{p}_k$ je možné vyjádřit ve tvaru

$$r_k = \mathcal{R}_k(\mathbf{A}) r_0, \qquad p_k = \mathcal{P}_k(\mathbf{A}) r_0,$$
$$\widetilde{r}_k = \mathcal{R}_k(\mathbf{A}^T) \widetilde{r}_0, \qquad \widetilde{p}_k = \mathcal{P}_k(\mathbf{A}^T) \widetilde{r}_0,$$

kde $\mathcal{R}_k(\mathbf{A})$ a $\mathcal{P}_k(\mathbf{A})$ jsou maticové polynomy stupně k. Ze vztahu (1.5) navíc plyne

$$\mathcal{R}_k \in \Pi_k,$$

kde Π_k označuje třídu polynomů stupně k s vlastností $\mathcal{R}(0) = 1$ (konstantní člen je roven jedné). Koeficienty $\boldsymbol{\alpha}_k$ a $\boldsymbol{\beta}_k$ lze potom psát následovně

$$\boldsymbol{\alpha}_k = \frac{(\mathcal{R}_k(\mathbf{A}^T)\widetilde{r}_0, \mathcal{R}_k(\mathbf{A})r_0)}{(\mathcal{P}_k(\mathbf{A}^T)\widetilde{r}_0, \mathbf{A}\mathcal{P}_k(\mathbf{A})r_0)}, \quad \boldsymbol{\beta}_k = \frac{(\mathcal{R}_{k+1}(\mathbf{A}^T)\widetilde{r}_0, \mathcal{R}_{k+1}(\mathbf{A})r_0)}{(\mathcal{R}_k(\mathbf{A}^T)\widetilde{r}_0, \mathcal{R}_k(\mathbf{A})r_0)}. \tag{1.109}$$

Protože platí

$$(\widetilde{p}_k, \mathbf{A}p_k) = (\widetilde{r}_k + \boldsymbol{\beta}_{k-1}p_{k-1}, \mathbf{A}p_k) = (\widetilde{r}_k, \mathbf{A}p_k),$$

lze zřejmě koeficient $\boldsymbol{\alpha}_k$ vyjádřit ve tvaru

$$\boldsymbol{\alpha}_k = \frac{(\widetilde{r}_k, r_k)}{(\widetilde{r}_k, \mathbf{A}p_k)} = \frac{(\mathcal{R}_k(\mathbf{A}^T)\widetilde{r}_0, r_k)}{(\mathcal{R}_k(\mathbf{A}^T)\widetilde{r}_0, \mathbf{A}p_k)}. \tag{1.110}$$

Uvědomíme-li si, že platí $r_k \perp \mathcal{K}_k(\mathbf{A}^T, \widetilde{r}_0)$ a $\mathbf{A}p_k \perp \mathcal{K}_k(\mathbf{A}^T, \widetilde{r}_0)$, lze polynom $\mathcal{R}_k(\xi)$ v (1.110) nahradit libovolným polynomem $\mathcal{Q}_k(\xi)$ stupně k, platí

$$\boldsymbol{\alpha}_k = \frac{(\mathcal{Q}_k(\mathbf{A}^T)\widetilde{r}_0, r_k)}{(\mathcal{Q}_k(\mathbf{A}^T)\widetilde{r}_0, \mathbf{A}p_k)} = \frac{(\widetilde{r}_0, \mathcal{Q}_k(\mathbf{A})r_k)}{(\widetilde{r}_0, \mathbf{A}\mathcal{Q}_k(\mathbf{A})p_k)}.$$

Využijeme-li vztahu

$$(\widetilde{r}_{k+1}, r_{k+1}) = (\widetilde{r}_k - \boldsymbol{\alpha}_k\mathbf{A}^T\widetilde{p}_k, r_{k+1}) = -\boldsymbol{\alpha}_k(\mathbf{A}^T\widetilde{p}_k, r_{k+1}) = -\frac{(\widetilde{r}_k, r_k)}{(\widetilde{p}_k, \mathbf{A}p_k)}(\mathbf{A}^T\widetilde{p}_k, r_{k+1})$$

a podmínek $r_{k+1} \perp \mathcal{K}_{k+1}(\mathbf{A}^T, \widetilde{r}_0)$ a $\mathbf{A}p_k \perp \mathcal{K}_k(\mathbf{A}^T, \widetilde{r}_0)$, lze koeficient $\boldsymbol{\beta}_k$ psát ve tvaru

$$\begin{aligned}
\boldsymbol{\beta}_k &= -\frac{(\mathbf{A}^T\widetilde{p}_k, r_{k+1})}{(\widetilde{p}_k, \mathbf{A}p_k)} = -\frac{(\mathbf{A}^T\mathcal{Q}_k(\mathbf{A}^T)\widetilde{r}_0, r_{k+1})}{(\mathcal{Q}_k(\mathbf{A}^T)\widetilde{r}_0, \mathbf{A}p_k)} = -\frac{(\mathcal{Q}_{k+1}(\mathbf{A}^T)\widetilde{r}_0, r_{k+1})}{(\mathcal{Q}_k(\mathbf{A}^T)\widetilde{r}_0, \mathbf{A}p_k)}\frac{q_k}{q_{k+1}} \\
&= -\frac{(\widetilde{r}_0, \mathcal{Q}_{k+1}(\mathbf{A})r_{k+1})}{(\widetilde{r}_0, \mathbf{A}\mathcal{Q}_k(\mathbf{A})p_k)}\frac{q_k}{q_{k+1}}, \qquad (1.111)
\end{aligned}$$

kde $\mathcal{Q}_k(\xi)$ a $\mathcal{Q}_{k+1}(\xi)$ jsou libovolné polynomy stupně k a $k+1$ a čísla q_k a q_{k+1} jsou nenulové vedoucí koeficienty polynomů $\mathcal{Q}_k(\xi)$ a $\mathcal{Q}_{k+1}(\xi)$.

Definujme nyní vektory

$$\mathbf{r}_k \equiv \mathcal{Q}_k\mathcal{R}_k(\mathbf{A})r_0, \quad \mathbf{p}_k \equiv \mathcal{Q}_k\mathcal{P}_k(\mathbf{A})r_0, \tag{1.112}$$

$\mathbf{x}_k$ vztahem $\mathbf{r}_k = b - \mathbf{A}\mathbf{x}_k$ a nechť $\mathcal{Q}_k \in \Pi_k$. Potom zřejmě platí

$$\mathcal{Q}_k\mathcal{R}_k \in \Pi_{2k}$$

a

$$\mathbf{x}_k \in \mathbf{x}_0 + \mathcal{K}_{2k}(\mathbf{A}, r_0), \qquad \mathbf{r}_k \in \mathbf{r}_0 + \mathbf{A}\mathcal{K}_{2k}(\mathbf{A}, r_0).$$

Čísla $\boldsymbol{\alpha}_k$ a $\boldsymbol{\beta}_k$ je možné vyjádřit s využitím vektorů $\mathbf{r}_k$ a $\mathbf{p}_k$,

$$\boldsymbol{\alpha}_k = \frac{(\widetilde{r}_0, \mathbf{r}_k)}{(\widetilde{r}_0, \mathbf{A}\mathbf{p}_k)}, \qquad \boldsymbol{\beta}_k = \frac{(\widetilde{r}_0, \mathbf{r}_{k+1})}{(\widetilde{r}_0, \mathbf{A}\mathbf{p}_k)}\frac{q_k}{q_{k+1}}. \tag{1.113}$$

Postupem uvedeným výše jsme se zřejmě vyhnuli násobení maticí $\mathbf{A}^T$ (v jistém smyslu jsme ho převedli na násobení maticí $\mathbf{A}$) a navíc máme možnost vhodně volit polynom

$\mathcal{Q}_k(\xi)$. Metody, jejichž rezidua lze vyjádřit ve tvaru (1.112), se nazývají *metody Lanczosova typu* (*Lanczos-Type Product Methods*) (LTPM) [30]. Algoritmy metod LTPM se formulují na základě znalostí rekurencí pro polynomy $\mathcal{R}_k(\xi)$ a $\mathcal{Q}_k(\xi)$. Pokud aplikujeme maticový polynom $\mathcal{Q}_k(\mathbf{A})$ na rekurenci pro vektor r_k, získáme rekurenci pro reziduum $\mathbf{r}_k$. V rekurenci pro $\mathbf{r}_k$ se vyskytnou i vektory, které nelze vyjádřit na základě definice (1.112) a které bude nutno vhodně pojmenovat a nalézt pro ně rekurentní vztahy.

Obecněji lze použít k výpočtu koeficientů $\boldsymbol{\alpha}_k$ a $\boldsymbol{\beta}_k$ polynom $\mathcal{U}_k(\xi)$ stupně k v součinovém tvaru

$$\mathcal{U}_k(\xi) \equiv \widetilde{\mathcal{Q}}_k \mathcal{Q}_k^{(d_k)}(\xi), \tag{1.114}$$

kde $\mathcal{Q}_k^{(d_k)} \in \Pi_{d_k}$, d_k je přirozené číslo, $d_k \leq k$, a $\widetilde{\mathcal{Q}}_k(\xi)$ je libovolný polynom stupně $k - d_k$. Rezidua $\mathbf{r}_k$ a vektory $\mathbf{p}_k$ potom uvažujeme ve tvaru

$$\mathbf{r}_k \equiv \mathcal{Q}_k^{(d_k)} \mathcal{R}_k(\mathbf{A}) r_0, \qquad \mathbf{p}_k \equiv \mathcal{Q}_k^{(d_k)} \mathcal{P}_k(\mathbf{A}) r_0 \tag{1.115}$$

a koeficienty počítáme podle vztahů

$$\boldsymbol{\alpha}_k = \frac{(\widetilde{r}_0, \mathcal{U}_k \mathcal{R}_k(\mathbf{A}) r_0)}{(\widetilde{r}_0, \mathbf{A}\mathcal{U}_k \mathcal{P}_k(\mathbf{A}) r_0)} = \frac{(\widetilde{\mathcal{Q}}_k(\mathbf{A}^T)\widetilde{r}_0, \mathbf{r}_k)}{(\widetilde{\mathcal{Q}}_k(\mathbf{A}^T)\widetilde{r}_0, \mathbf{A}\mathbf{p}_k)}, \tag{1.116}$$

$$\boldsymbol{\beta}_k = \frac{(\widetilde{r}_0, \mathcal{U}_{k+1} \mathcal{R}_{k+1}(\mathbf{A}) r_0)}{(\widetilde{r}_0, \mathbf{A}\mathcal{U}_k \mathcal{P}_k(\mathbf{A}) r_0)} \frac{u_k}{u_{k+1}} = \frac{(\widetilde{\mathcal{Q}}_{k+1}(\mathbf{A}^T)\widetilde{r}_0, \mathbf{r}_{k+1})}{(\widetilde{\mathcal{Q}}_k(\mathbf{A}^T)\widetilde{r}_0, \mathbf{A}\mathbf{p}_k)} \frac{u_k}{u_{k+1}}, \tag{1.117}$$

kde u_k a u_{k+1} jsou nenulové vedoucí koeficienty polynomů $\mathcal{U}_k(\xi)$ a $\mathcal{U}_{k+1}(\xi)$. Metody, jejichž rezidua a lze vyjádřit ve tvaru (1.115) nazýváme *hybridní* BiCG-*metody* (*Hybrid* BiCG-*Methods*), viz. [54].

1.5.1 Metoda CGS

Volbou $\mathcal{Q}_k(\xi) \equiv \mathcal{R}_k(\xi)$ definujeme *umocněnou metodu sdružených gradientů* CGS (*Conjugate Gradient Square Method*) [55]. Rezidua této metody je možné vyjádřit ve tvaru

$$\mathbf{r}_k^{CGS} \equiv \mathcal{R}_k^2(\mathbf{A}) r_0.$$

Odvození algoritmu metody CGS z BiCG je pouze technickým problémem. Toto odvození lze nalézt například v [61] nebo v [30].

Chování a vlastnosti metody CGS lze stručně shrnout následovně. Pokud nalezne BiCG řešení ($\mathbf{o} = r_k = \mathcal{R}_k(\mathbf{A}) r_0$), nalezne ho i metoda CGS. K předčasnému ukončení algoritmu CGS (nelze vypočíst koeficienty $\boldsymbol{\alpha}_k$ či $\boldsymbol{\beta}_k$) dochází právě tehdy, dojde-li k předčasnému ukončení algoritmu BiCG. Jestliže během výpočtu BiCG dochází k rychlé konvergenci, lze u CGS očekávat velké zrychlení konvergence a nalezení dobré aproximace řešení v podstatně méně iteracích než u BiCG. Pokud dochází při běhu BiCG k převýšením v reziduové křivce, lze očekávat tatáž převýšení i v reziduové křivce CGS, ovšem s podstatně větší výchylkou. Důsledkem toho se může při aplikaci algoritmu CGS v konečné aritmetice počítače velmi snížit hladina limitní přesnosti s níž jsme schopni vypočíst aproximaci řešení [23].

Modifikací metody CGS je zhlazená metoda CGS [67], která navíc počítá dva páry vektorů, zhlazená rezidua a příslušné aproximace. Zhlazení s sebou sice přináší monotónní reziduovou křivku (viz. např. [61]), tedy bez převýšení, nicméně lze očekávat, že všechny nestability při výpočtu algoritmu CGS se projeví i při výpočtu algoritmu zhlazené metody CGS. Příčina spočívá v tom, že se rezidua a aproximace řešení zhlazené metody CGS počítají z původních reziduí a aproximací řešení algoritmu metody CGS. Silný nástroj na řešení soustav dělá z CGS resp. ze zhlazené CGS teprve až předpodmínění.

Algoritmus 1.7. CGS

input $x_0,\ \mathbf{A},\ b,\ \widetilde{r}_0$ **for** $k = 0,\ \ldots$

initialization

$$\begin{aligned} \mathbf{r}_0 &= b - \mathbf{A}\mathbf{x}_0 \\ \mathbf{p}_0 &= \mathbf{r}_0 \\ u_0 &= \mathbf{r}_0 \end{aligned}$$

$$\begin{aligned} \boldsymbol{\alpha}_k &= \frac{\widetilde{r}_0^T \mathbf{r}_k}{\widetilde{r}_0^T \mathbf{A}\mathbf{p}_k} \\ q_k &= u_k - \boldsymbol{\alpha}_k \mathbf{A}\mathbf{p}_k \\ \mathbf{r}_{k+1} &= \mathbf{r}_k - \boldsymbol{\alpha}_k \mathbf{A}(u_k + q_k) \\ \mathbf{x}_{k+1} &= \mathbf{x}_k + \boldsymbol{\alpha}_k (u_k + q_k) \\ \boldsymbol{\beta}_k &= \frac{\widetilde{r}_0^T \mathbf{r}_{k+1}}{\widetilde{r}_0^T \mathbf{r}_k} \\ u_{k+1} &= \mathbf{r}_{k+1} + \boldsymbol{\beta}_k q_k \\ \mathbf{p}_{k+1} &= u_{k+1} + \boldsymbol{\beta}_k (q_k + \boldsymbol{\beta}_k \mathbf{p}_k) \end{aligned}$$

end for

1.5.2 Metoda BiCGStab

Reziduové polynomy $\mathcal{R}_k(\xi)$ Lanczosovy metody pro řešení systému lineárních rovnic (určené algoritmem BiCG) lze podle úvodního odstavce sekce 1.5 kombinovat s libovolnými polynomy $\mathcal{Q}_k(\xi)$ z Π_k. Posloupnost těchto polynomů může být určena jakýmkoliv předpisem, který zajišťuje, že platí $\mathcal{Q}_k \in \Pi_k$. Pokud však požadujeme, aby byl výsledný algoritmus metody paměťově nenáročný a laciný, co se počtu operací na jednu iteraci týče, je vhodné uvažovat vytváření polynomů $\mathcal{Q}_k(\xi)$ krátkou rekurencí. Jako příklad uveďme dvoukrokovou rekurenci

$$\mathcal{Q}_{k+1}(\xi) = (1 - \chi_k \xi)\mathcal{Q}_k(\xi), \quad \mathcal{Q}_0 \equiv 1, \tag{1.118}$$

jež je volena tak, aby při libovolné volbě koeficientu $\chi_k \neq 0$, byla automaticky splněna podmínka $\mathcal{Q}_{k+1} \in \Pi_{k+1}$. Zvolíme-li koeficient χ_k tak, že má reziduum

$$\mathbf{r}_{k+1}^{Stab} \equiv \mathcal{Q}_{k+1}\mathcal{R}_{k+1}(\mathbf{A})r_0 = (\mathbf{I} - \chi_k \mathbf{A})\mathcal{Q}_k \mathcal{R}_{k+1}(\mathbf{A})r_0$$

minimální normu, získáme metodu zvanou BiCGStab [64]. Odvození algoritmu metody BiCGStab ze znalosti rekurencí pro oba polynomy je možné najít např. v [30]. Protože je vedoucí koeficient q_k polynomu $\mathcal{Q}_k(\xi)$ roven

$$q_k = \prod_{i=0}^{k-1} \chi_i,$$

je zřejmě $q_k/q_{k+1} = \chi_k^{-1}$, a podle (1.113) je

$$\boldsymbol{\beta}_k = \chi_k^{-1} \frac{\widetilde{r}_0^T \mathbf{r}_{k+1}}{\widetilde{r}_0^T \mathbf{A}\mathbf{p}_k}.$$

Je zřejmé, že v algoritmu metody BiCGStab může oproti BiCG navíc dojít k předčasnému ukončení výpočtu, je-li $\chi_k = 0$ (minimalizační předčasné ukončení). Pokud je χ_k blízké nule, může dojít v konečné aritmetice počítače k velmi nepřesnému vypočtení koeficientu $\boldsymbol{\beta}_k$, ke znehodnocení výpočtu a další chování algoritmu je nepředvídatelné. Naneštěstí existují aplikace, ve kterých má χ_k tendenci být malé. Je to především v případech, kdy

Algoritmus 1.8. BiCGStab

input $x_0,\ \mathbf{A},\ b,\ \widetilde{r}_0$

initialization

$$\begin{aligned} \mathbf{r}_0 &= b - \mathbf{A}\mathbf{x}_0 \\ \mathbf{p}_0 &= \mathbf{r}_0 \end{aligned}$$

for $k = 0,\ \ldots$

$$\begin{aligned} \boldsymbol{\alpha}_k &= \frac{\widetilde{r}_0^T \mathbf{r}_k}{\widetilde{r}_0^T \mathbf{A}\mathbf{p}_k} \\ v_k &= \mathbf{r}_k - \boldsymbol{\alpha}_k \mathbf{A} p_k \\ \chi_k &= \frac{v_k^T \mathbf{A} v_k}{\|\mathbf{A}\ v_k\|^2} \\ \mathbf{r}_{k+1} &= v_k - \chi_k \mathbf{A} v_k \\ \mathbf{x}_{k+1} &= \mathbf{x}_k + \boldsymbol{\alpha}_k p_k + \chi_k v_k \\ \boldsymbol{\beta}_k &= \chi_k^{-1} \frac{\widetilde{r}_0^T r_{k+1}}{\widetilde{r}_0^T \mathbf{A} p_k} \\ \mathbf{p}_{k+1} &= \mathbf{r}_{k+1} + \boldsymbol{\beta}_k (\mathbf{p}_k - \chi_k \mathbf{A}\mathbf{p}_k) \end{aligned}$$

end for

má matice $\mathbf{A}$ vlastní čísla, jež mají malou reálnou část a nemalou imaginární část. Příčina tkví v tom, že pro reálnou matici $\mathbf{A}$ a reálnou pravou stranu b jsou kořeny polynomu $\mathcal{Q}_k(\xi)$ (rovny $1/\chi_i$) reálné a tudíž tento polynom nemůže účinně působit při tlumení chybových komponent příslušných k vlastním číslům matice $\mathbf{A}$, jež mají velkou imaginární část. Pokusme se uvedenou skutečnost vyjádřit intuitivní úvahou.

Uvažujme Jordanův kanonický tvar matice $\mathbf{A}$, $\mathbf{A} = \mathbf{C}\mathbf{J}\mathbf{C}^{-1}$. Potom můžeme reziduum metody BiCGStab vyjádřit ve tvaru

$$\mathbf{r}_k = \mathcal{Q}_k\mathcal{R}_k(\mathbf{C}\mathbf{J}\mathbf{C}^{-1})r_0 = \mathbf{C}\begin{pmatrix} \mathcal{Q}_k\mathcal{R}_k(\mathbf{J}_1) & 0 & 0 \\ 0 & \mathcal{Q}_k\mathcal{R}_k(\mathbf{J}_2) & 0 \\ 0 & 0 & \mathcal{Q}_k\mathcal{R}_k(\mathbf{J}_j) \end{pmatrix}\mathbf{C}^{-1}r_0.$$

Polynom $\mathcal{Q}_k(\xi)$ přispívá k tomu, aby měly matice $\mathcal{Q}_k\mathcal{R}_k(\mathbf{J}_i)$ co nejmenší prvky. Má-li však Jordanova buňka $\mathbf{J}_i$ na diagonále vlastní číslo, jež má reálnou část malou a imaginární velkou, může se stát, že polynom $\mathcal{Q}_k(\xi)$ s reálnými kořeny nejen že netlumí velikost prvků v matici $\mathcal{Q}_k\mathcal{R}_k(\mathbf{J}_i)$, ale naopak způsobí, že prvky v této matici jsou hodně velké.

I přes možnost předčasného ukončení algoritmu BiCGStab z důvodu nulovosti koeficientu χ_k je BiCGStab v praxi jedna z nejpoužívanějších krylovovských metod. Důvodem je ve většině případů podstatně rychlejší konvergence než u BiCG.

1.5.3 Metody BiCGStab2 a BiCG×MR2

K určení polynomů $\mathcal{Q}_{k+1}(\xi)$ lze samozřejmě použít i tří-krokových rekurencí, čímž získáme více stupňů volnosti. Uvažujme rekurenci

$$\mathcal{Q}_{k+1}(\xi) = (\gamma_k + \delta_k\xi)\mathcal{Q}_k(\xi) + (1-\gamma_k)\mathcal{Q}_{k-1}(\xi), \quad \mathcal{Q}_0 \equiv 1, \quad \mathcal{Q}_{-1} \equiv 0. \tag{1.119}$$

Koeficienty $\delta_k \neq 0$ a γ_k lze volit libovolně a z (1.119) plyne, že $\mathcal{Q}_{k+1} \in \Pi_{k+1}$.

Za polynomy $\{\mathcal{Q}_i(\xi)\}_{i=0}^{k+1}$ je možné volit např. vhodně posunuté Čebyševovy polynomy nebo, jako v případě BiCGStab, je možné určit koeficienty γ_k a δ_k tak, aby mělo výsledné reziduum

$$\mathbf{r}_{k+1} \equiv \mathcal{Q}_{k+1}\mathcal{R}_{k+1}(\mathbf{A})r_0 = \big((\gamma_k + \delta_k\xi)\mathcal{Q}_k(\xi) + (1-\gamma_k)\mathcal{Q}_{k-1}(\xi)\big)\mathcal{R}_{k+1}(\mathbf{A})r_0$$

minimální normu. Získáme tak metodu zvanou BiCG×MR2 ([30]). Kombinováním dvou a tříkrokových rekurencí pro počítání polynomu $\mathcal{Q}_{k+1}(\xi)$ s volbou koeficientů vedoucí k minimální normě výsledného rezidua, obdržíme metodu zvanou BiCGStab2. Algoritmy obou metod lze nalézt v [30].

1.5.4 Metoda BiCGStab(l)

Klasickou hybridní BiCG-metodou je BiCGStab(l) [54]. Nechť $k = ml$. Pro vypočtení koeficientů $\boldsymbol{\alpha}_{k+j}$ a $\boldsymbol{\beta}_{k+j-1}$ $(j = 0, \ldots, l-1)$ použijeme polynom

$$\mathcal{U}_{k+j}(\mathbf{A}) \equiv \mathbf{A}^j \overbrace{\mathcal{S}_{m-1} \ldots \mathcal{S}_0(\mathbf{A})}^{\mathcal{Q}_{k+j}^{(k)}},$$

kde $\mathcal{S}_i(\xi)$ jsou polynomy stupně l. Zřejmě tedy v (1.114) volíme $\widetilde{\mathcal{Q}}_{k+j}(\mathbf{A}) \equiv \mathbf{A}^j$ a $\mathcal{Q}_{k+j}^{(k)}(\mathbf{A}) \equiv \mathcal{Q}_k^{(k)}(\mathbf{A}) \equiv \mathcal{S}_{m-1} \ldots \mathcal{S}_0(\mathbf{A})$. Reziduum $\mathbf{r}_{k+j}^{Stab(l)}$ je definováno jako

$$\mathbf{r}_{k+j}^{Stab(l)} \equiv \mathcal{Q}_{k+j}^{(k)} \mathcal{R}_{k+j}(\mathbf{A}) r_0 = \mathcal{S}_{m-1} \ldots \mathcal{S}_0 \mathcal{R}_{k+j}(\mathbf{A}) r_0$$

a polynom $\mathcal{S}_{m-1}(\xi)$ vztahem

$$\mathcal{S}_{m-1}(\xi) \equiv \underset{\mathcal{S} \in \Pi_l}{\arg\min} \|\mathcal{S} \overbrace{\mathcal{S}_{m-2} \ldots \mathcal{S}_0}^{\mathcal{Q}_{k-l}^{(k-l)}} \mathcal{R}_k(\mathbf{A}) r_0\|.$$

Polynom $\mathcal{S}_{m-1}(\xi)$ je zřejmě volen tak, aby mělo reziduum $\mathbf{r}_k$ minimální normu. Metodu BiCGStab(l) můžeme v jistém smyslu považovat za kombinaci BiCG s principy metody GMRES aplikované na l vektorů. Podrobněji, výpočet probíhá ve dvou částech

1. Podle rekurence, která vznikne aplikací maticového polynomu $\mathcal{Q}_{k-l}(\mathbf{A})$ na rekurence algoritmu BiCG vypočteme vektory $\hat{r}_0 \equiv \mathcal{Q}_{k-l}\mathcal{R}_k(\mathbf{A})r_0$, $\mathbf{A}\hat{r}_0, \ldots, \mathbf{A}^l\hat{r}_0$.

2. Najdeme koeficienty polynomu $\mathcal{S}_{m-1}(\xi) \equiv 1 - \sum_{i=1}^{l} \sigma_i \xi^i$ tak, že minimalizujeme výraz

$$\|\mathcal{S}_{m-1}\mathcal{Q}_{k-l}\mathcal{R}_k(\mathbf{A})r_0\| = \|\hat{r}_0 - \sum_{i=1}^{l} \sigma_i \mathbf{A}^i \hat{r}_0\|$$

   přes všechny možné σ_i.

1.5.5 Kvazi-minimální pojetí metody CGS (TFQMR)

Podstatnou nevýhodou metody CGS, jak jsme se zmínili v odstavci 1.5.1, jsou často vznikající velká převýšení v reziduové křivce, snížení dosažitelné limitní hladiny přesnosti a možné znehodnocení výpočtu v důsledku nepřesného vypočtení vektorů a koeficientů v konečné aritmetice počítače. Samotný princip umocnění, t.j. místo vektorů $r_k = \mathcal{R}_k(\mathbf{A})r_0$ počítat vektory $\mathbf{r}_k = \mathcal{R}_k^2(\mathbf{A})r_0$, nám však umožnil vyhnout se počítání s traspozicí matice $\mathbf{A}$ a počítat s téměř stejnými výpočetními náklady v k iteracích aproximace řešení z variety $x_0 + \mathcal{K}_{2k}(\mathbf{A}, r_0)$. Je tedy na místě se ptát, zda nelze umocnění aplikovat na reziduové polynomy nějaké jiné krylovovské metody, jejíž reziduová konvergenční křivka má hladší průběh než reziduová konvergenční křivka BiCG. Nabízí se metoda QMR. Do současné doby však nebyl nalezen algoritmus, který by byl schopen počítat umocněná rezidua metody QMR s přibližně stejnými výpočetními náklady jako QMR. Doposud známé algorimy

počítající umocněná rezidua QMR s použitím tří násobení matice-vektor v jednom kroku a tím pozbývají lacinosti, co se počtu operací na jednu iteraci týče. V hledání krylovovské metody s hladkou konvergenční křivkou založené na umocněných reziduích byl úspěšný až Freund [12]. Z vektorů vystupujících v algoritmu metody CGS vybral vhodnou bázi a nalezl algoritmus, který počítá rezidua s minimálním souřadnicovým vektorem v této bázi. Toto QMR-pojetí aplikované na vektory metody CGS nazval TFQMR (*Transpose Free* QMR). Ukažme si v dalším odvození algoritmu této metody.

Algoritmus 1.7 metody CGS modifikujme následujícím způsobem. Ostříškujme všechny vektory a skaláry vyskytující se v algoritmu 1.7 a přenásobme jejich indexy číslem 2, definujeme tedy např. $\widehat{\mathbf{x}}_{2k} \equiv \mathbf{x}_k$, $\widehat{\boldsymbol{\alpha}}_{2k} \equiv \boldsymbol{\alpha}_k$ atd. Na poslední vzniklou rekurenci $\widehat{p}_{2k+2} = \widehat{u}_{2k+2} + \widehat{\boldsymbol{\beta}}_{2k}(\widehat{q}_{2k} + \widehat{\boldsymbol{\beta}}\widehat{p}_{2k})$ aplikujme matici $\mathbf{A}$ a budeme psát v celém algoritmu $\widehat{\mathbf{p}}_{2k}$ místo $\mathbf{A}\widehat{p}_{2k}$. Definujeme-li nyní vektory $\widehat{\mathbf{s}}_{2k}$ a $\widehat{\mathbf{s}}_{2k+1}$ jako $\widehat{\mathbf{s}}_{2k} \equiv \widehat{u}_{2k}$ a $\widehat{\mathbf{s}}_{2k+1} \equiv \widehat{q}_{2k}$, „meziaproximaci“ $\widehat{\mathbf{x}}_{2k+1}$ s příslušným reziduem vztahy $\widehat{\mathbf{x}}_{2k+1} \equiv \widehat{\mathbf{x}}_{2k} + \widehat{\boldsymbol{\alpha}}_{2k}\widehat{\mathbf{s}}_{2k}$, $\widehat{\mathbf{r}}_{2k+1} \equiv \widehat{\mathbf{r}}_{2k} - \widehat{\boldsymbol{\alpha}}_{2k}\mathbf{A}\widehat{\mathbf{s}}_{2k}$ a skalár $\widehat{\boldsymbol{\alpha}}_{2k+1}$ jako $\widehat{\boldsymbol{\alpha}}_{2k+1} \equiv \widehat{\boldsymbol{\alpha}}_{2k}$, můžeme výsledný modifikovaný algoritmus metody CGS psát ve tvaru algoritmu 1.9.

ALGORITMUS 1.9. *Modifikovaná* CGS

input x_0, $\mathbf{A}$, b, $\widetilde{r}_0$

initialization

$$\begin{aligned} r_0 &= b - \mathbf{A}x_0 \\ \widehat{\mathbf{x}}_0 &= x_0 \\ \widehat{\mathbf{r}}_0 &= r_0 \\ \widehat{\mathbf{s}}_0 &= r_0 \\ \widehat{\mathbf{p}}_0 &= \mathbf{A}r_0 \end{aligned}$$

for $k = 0, \ldots$

$$\begin{aligned} \widehat{\boldsymbol{\alpha}}_{2k+1} = \widehat{\boldsymbol{\alpha}}_{2k} &= \frac{\widetilde{r}_0^T \widehat{\mathbf{r}}_{2k}}{\widetilde{r}_0^T \widehat{\mathbf{p}}_{2k}} \\ \widehat{\mathbf{s}}_{2k+1} &= \widehat{\mathbf{s}}_{2k} - \widehat{\boldsymbol{\alpha}}_{2k}\widehat{\mathbf{p}}_{2k} \\ \widehat{\mathbf{r}}_{2k+1} &= \widehat{\mathbf{r}}_{2k} - \widehat{\boldsymbol{\alpha}}_{2k}\mathbf{A}\widehat{\mathbf{s}}_{2k} \\ \widehat{\mathbf{x}}_{2k+1} &= \widehat{\mathbf{x}}_{2k} + \widehat{\boldsymbol{\alpha}}_{2k}\widehat{\mathbf{s}}_{2k} \\ \widehat{\mathbf{r}}_{2k+2} &= \widehat{\mathbf{r}}_{2k+1} - \widehat{\boldsymbol{\alpha}}_{2k+1}\mathbf{A}\widehat{\mathbf{s}}_{2k+1} \\ \widehat{\mathbf{x}}_{2k+2} &= \widehat{\mathbf{x}}_{2k+2} + \widehat{\boldsymbol{\alpha}}_{2k+1}\widehat{\mathbf{s}}_{2k+1} \\ \widehat{\boldsymbol{\beta}}_{2k} &= \frac{\widetilde{r}_0^T \widehat{\mathbf{r}}_{2k+2}}{\widetilde{r}_0^T \widehat{\mathbf{r}}_{2k}} \\ \widehat{\mathbf{s}}_{2k+2} &= \widehat{\mathbf{r}}_{2k+2} + \widehat{\boldsymbol{\beta}}_{2k}\widehat{\mathbf{s}}_{2k+1} \\ \widehat{\mathbf{p}}_{2k+2} &= \mathbf{A}\widehat{\mathbf{s}}_{2k+2} + \widehat{\boldsymbol{\beta}}_{2k}(\mathbf{A}\widehat{\mathbf{s}}_{2k+1} + \widehat{\boldsymbol{\beta}}_{2k}\widehat{\mathbf{p}}_{2k}) \end{aligned}$$

end for

Definujme matice $\widehat{\mathbf{S}}_k$ a $\widehat{\mathbf{R}}_k$ vztahy

$$\widehat{\mathbf{S}}_k \equiv [\widehat{\mathbf{s}}_0, \ldots, \widehat{\mathbf{s}}_{k-1}], \qquad \widehat{\mathbf{R}}_k \equiv [\widehat{\mathbf{r}}_0, \ldots, \widehat{\mathbf{r}}_{k-1}].$$

Zřejmě platí

$$\begin{aligned} \widehat{\mathbf{x}}_k &= \widehat{\mathbf{x}}_0 + \widehat{\mathbf{S}}_k z_k = \widehat{\mathbf{x}}_{k-1} + \widehat{\boldsymbol{\alpha}}_{k-1}\widehat{\mathbf{s}}_{k-1}, \\ \widehat{\mathbf{r}}_k &= \widehat{\mathbf{r}}_0 - \mathbf{A}\widehat{\mathbf{S}}_k z_k = \widehat{\mathbf{r}}_{k-1} - \widehat{\boldsymbol{\alpha}}_{k-1}\mathbf{A}\widehat{\mathbf{s}}_{k-1}, \end{aligned}$$

kde $z_k \in \mathbb{R}^k$, $z_k \equiv (\widehat{\boldsymbol{\alpha}}_0, \ldots, \widehat{\boldsymbol{\alpha}}_{k-1})^T$. Ze vztahu mezi rezidui $\widehat{\mathbf{r}}_k$ a $\widehat{\mathbf{r}}_{k-1}$ potom plyne $\widehat{\boldsymbol{\alpha}}_{k-1}^{-1}(\widehat{\mathbf{r}}_{k-1} - \widehat{\mathbf{r}}_k) = \mathbf{A}\widehat{\mathbf{s}}_{k-1}$, maticově zapsáno

$$\mathbf{A}\widehat{\mathbf{S}}_k = \widehat{\mathbf{R}}_{k+1}\underline{\mathbf{B}}_k, \tag{1.120}$$

kde $\underline{\mathbf{B}}_k \in \mathbb{R}^{(k+1)\times k}$ je matice

$$\underline{\mathbf{B}}_k = \begin{pmatrix} 1 & & & \\ -1 & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ & & & -1 \end{pmatrix} \begin{pmatrix} \widehat{\boldsymbol{\alpha}}_0^{-1} & & & \\ & \widehat{\boldsymbol{\alpha}}_1^{-1} & & \\ & & \ddots & \\ & & & \widehat{\boldsymbol{\alpha}}_{m-1}^{-1} \end{pmatrix}.$$

Vztahu (1.120) využijeme k vyjádření souřadnicového vektoru reziduí modifikované CGS. Zaveďme diagonální matici $\widehat{\mathbf{D}}_{k+1} \equiv \mathrm{diag}(\hat{d}_1, \ldots, \hat{d}_{k+1})$, kde $\hat{d}_i \neq 0$ jsou libovolná čísla $(i > 1)$ a $\hat{d}_1 \equiv \|r_0\|$. Definujme matice $\mathbf{V}_{k+1} \equiv \widehat{\mathbf{R}}_{k+1}\widehat{\mathbf{D}}_{k+1}^{-1}$ a $\underline{\mathbf{H}}_k \equiv \widehat{\mathbf{D}}_{k+1}\underline{\mathbf{B}}_k$. Sloupce matice $\mathbf{V}_{k+1}$ jsou zřejmě reziduové vektory modifikované CGS normované pomocí čísel $\hat{d}_1, \ldots, \hat{d}_{k+1}$. Reziduum $\widehat{\mathbf{r}}_k$ potom můžeme vyjádřit ve tvaru

$$\begin{aligned} \widehat{\mathbf{r}}_k &= \widehat{\mathbf{r}}_0 - \mathbf{A}\widehat{\mathbf{S}}_k z_k = \widehat{\mathbf{R}}_{k+1}\widehat{\mathbf{D}}_{k+1}^{-1}\widehat{\mathbf{D}}_{k+1}(e_1^{(k+1)} - \underline{\mathbf{B}}_k z_k) \\ &= \mathbf{V}_{k+1}(\|r_0\| e_1^{(k+1)} - \underline{\mathbf{H}}_k z_k). \end{aligned} \tag{1.121}$$

Podle (1.70) jsou $\widehat{\mathbf{r}}_k$ rezidua s galerkinovskými kvazi-rezidui a lze zřejmě formulovat metodu, jejíž rezidua budou mít v bázi $\widehat{\mathbf{s}}_0, \ldots, \widehat{\mathbf{s}}_{k-1}$ minimální normu souřadnicového vektoru, t.j. metodu s minimálními kvazi-rezidui. Tuto metodu nazveme TFQMR (*Traspose Free Quasi Minimal Residual Method*).

K odvození algoritmu metody TFQMR použijeme výsledků z odstavce 1.3. Označíme-li vektor z_k, který je argumentem minima normy souřadnicového vektoru, jako z_k^M, je k-tá aproximace řešení metody TFQMR určena vztahem

$$x_k^{TFQMR} = x_0 + \widehat{\mathbf{S}}_k z_k^M.$$

Z výsledků odstavce 1.3 plynou vztahy

$$\begin{aligned} \eta_1 &= \hat{d}_1, \\ \eta_{k+1}^2 &= \frac{\eta_k^2 \|\widehat{\mathbf{r}}_k\|^2}{\eta_k^2 + \|\widehat{\mathbf{r}}_k\|^2}, \\ c_k^2 &= \frac{\eta_{k+1}^2}{\|\widehat{\mathbf{r}}_k\|^2}, \\ s_k^2 &= 1 - c_k^2, \\ \mathbf{r}_k^{TFQMR} &= s_k^2 \mathbf{r}_{k-1}^{TFQMR} + c_k^2 \widehat{\mathbf{r}}_k, \\ \mathbf{x}_k^{TFQMR} &= s_k^2 \mathbf{x}_{k-1}^{TFQMR} + c_k^2 \widehat{\mathbf{x}}_k. \end{aligned}$$

Pozměňme algoritmus, který vznikne složením právě uvedených vztahů a rekurencí pro modifikovanou metodu CGS tak, že rozdělíme dvojkrok (počítáme dvě rezidua v jednom kroku) na dva samostatné kroky. Výsledkem je algoritmus 1.10, ve kterém píšeme r_k a x_k místo $\mathbf{r}_k^{TFQMR}$ a $\mathbf{x}_k^{TFQMR}$.

1.6 Zastavovací kritérium

Jak jsme se zmínili v úvodu této práce, jednou z velkých výhod iteračních metod je skutečnost, že v každé iteraci máme k dispozici aproximaci řešení soustavy. Iterační proces můžeme kdykoliv zastavit a vypočtenou aproximaci řešení považovat za přibližné řešení

Algoritmus 1.10. TFQMR

input $x_0,\ \mathbf{A},\ b,\ \widetilde{r}_0$

initialization

$$
\begin{aligned}
r_0 &= b - \mathbf{A}x_0 \\
\widehat{\mathbf{x}}_0 &= x_0 \\
\widehat{\mathbf{r}}_0 &= r_0 \\
\widehat{\mathbf{s}}_0 &= r_0 \\
\widehat{\mathbf{p}}_0 &= \mathbf{A}\widehat{\mathbf{r}}_0 \\
&\eta_1 = \|r_0\|
\end{aligned}
$$

for $k = 0,\ \ldots$

if k **is** *even* **then**

$$
\begin{aligned}
&\widehat{\boldsymbol{\alpha}}_{k+1} = \widehat{\boldsymbol{\alpha}}_k = \frac{\widetilde{r}_0^T \widehat{\mathbf{r}}_k}{\widetilde{r}_0^T \widehat{\mathbf{p}}_k} \\
\widehat{\mathbf{s}}_{k+1} &= \widehat{\mathbf{s}}_k - \widehat{\boldsymbol{\alpha}}_k \widehat{\mathbf{p}}_k
\end{aligned}
$$

end if

$$
\begin{aligned}
\widehat{\mathbf{r}}_{k+1} &= \widehat{\mathbf{r}}_k - \widehat{\boldsymbol{\alpha}}_k \mathbf{A}\widehat{\mathbf{s}}_k \\
\widehat{\mathbf{x}}_{k+1} &= \widehat{\mathbf{x}}_k + \widehat{\boldsymbol{\alpha}}_k \widehat{\mathbf{s}}_k \\
&\eta_{k+2}^2 = \frac{\eta_{k+1}^2 \|\widehat{\mathbf{r}}_{k+1}\|^2}{\eta_{k+1}^2 + \|\widehat{\mathbf{r}}_{k+1}\|^2} \\
&c_{k+1}^2 = \frac{\eta_{k+2}^2}{\|\widehat{\mathbf{r}}_{k+1}\|^2} \\
&s_{k+1}^2 = 1 - c_{k+1}^2 \\
r_{k+1} &= s_{k+1}^2 r_k + c_{k+1}^2 \widehat{\mathbf{r}}_{k+1} \\
x_{k+1} &= s_{k+1}^2 x_k + c_{k+1}^2 \widehat{\mathbf{x}}_{k+1}
\end{aligned}
$$

if k **is** *odd* **then**

$$
\begin{aligned}
&\widehat{\boldsymbol{\beta}}_k = \frac{\widetilde{r}_0^T \widehat{\mathbf{r}}_{k+1}}{\widetilde{r}_0^T \widehat{\mathbf{r}}_{k-1}} \\
\widehat{\mathbf{s}}_{k+1} &= \widehat{\mathbf{r}}_{k+1} + \widehat{\boldsymbol{\beta}}_k \widehat{\mathbf{s}}_k \\
\widehat{\mathbf{p}}_{k+1} &= \mathbf{A}\widehat{\mathbf{s}}_{k+1} + \widehat{\boldsymbol{\beta}}_k (\mathbf{A}\widehat{\mathbf{s}}_k + \widehat{\boldsymbol{\beta}}_k \widehat{\mathbf{p}}_{k-1})
\end{aligned}
$$

end if
end for

soustavy. Není však zřejmé, jak měřit kvalitu vypočtené aproximace řešení t.j. její „blízkost“ k řešení soustavy a kdy iterační proces zastavit. Ideální by zřejmě bylo, kdybychom byli schopni vypočíst hodnotu

$$|||x - x_k|||, \tag{1.122}$$

kde $|||\cdot|||$ označuje vektorovou normu. Volba této vektorové normy přitom může záviset na typu úlohy. Například některé fyzikální úlohy a úlohy kvantové chemie vedou na řešení soustavy lineárních rovnic se symetrickou, pozitivně definitní maticí a je přirozené měřit blízkost aproximace řešení k řešení soustavy pomocí $\mathbf{A}$-normy chyby (energetické normy)

$$\|x - x_k\|_{\mathbf{A}} \equiv (x - x_k, \mathbf{A}(x - x_k))^{1/2}.$$

Hodnoty $\mathbf{A}$-normy chyby nelze obecně vyjádřit přesně. Pokud však použijeme k řešení systému lineárních rovnic metodu sdružených gradientů (CG), lze velikost $\mathbf{A}$-normy chyby efektivně odhadnout a odhadu poté použít k zastavení algoritmu, viz. kapitola 3. Poznamenejme, že při výpočtu algoritmu CG je možné odhadovat i euklidovskou normu chyby.

V obecně nesymetrickém případě není známo, jak efektivně odhadnout normu chyby.

Informaci o konvergenci se proto snažíme získat z lehce dostupných reziduových vektorů

$$r_k = b - \mathbf{A}x_k.$$

Uvažujme, že soustava lineárních rovnic vznikla z matematického modelu po použití diskretizace. Potom jsou prvky matice $\mathbf{A}$ i prvky vektoru pravé strany zřejmě zatíženy chybou modelu a chybou diskretizace. Přesné řešení soustavy $\mathbf{A}x = b$ reprezentuje aproximaci řešení původní úlohy reálného světa a její kvalita je dána chybou modelu a chybou diskretizace. To ovšem znamená, že stejně „dobrou“ aproximací řešení původního problému reálného světa bude i řešení porušené soustavy

$$(\mathbf{A} + \Delta\mathbf{A})\widetilde{x} = b + \Delta b, \tag{1.123}$$

kde $\Delta\mathbf{A}$ a Δb představují malé poruchy vzhledem k chybě modelu a chybě diskretizace. V tomto smyslu lze definovat třídu soustav $\widetilde{\mathbf{A}}\widetilde{x} = \widetilde{b}$, které jsou blízké soustavě $\mathbf{A}x = b$ (ve smyslu malých poruch $\Delta\mathbf{A}$ a Δb) a jejichž řešení jsou stejně „dobré“ aproximace řešení původního problému reálného světa, jako řešení soustavy $\mathbf{A}x = b$.

Zřejmě bychom během iteračního procesu chtěli mít informaci o tom, jak velké musí být nejmenší poruchy $\Delta\mathbf{A}$ v matici $\mathbf{A}$ a Δb ve vektoru pravé strany b, aby právě vypočtená aproximace řešení x_k soustavy $\mathbf{A}x = b$ byla řešením porušené soustavy (1.123). Pokud jsou velikosti těchto poruch menší než požadovaná hranice určená na základě znalosti chyby modelu a chyby diskretizace, nemá již dále smysl pokračovat v iteračním procesu (přesnější aproximace řešení soustavy $\mathbf{A}x = b$ už není přesnější aproximací řešení původního problému reálného světa). Velikosti nejmenších poruch původní soustavy při nichž je vypočtená aproximace přesným řešením porušené soustavy lze měřit pomocí tkzv. *normované relativní zpětné chyby* $\beta(x_k)$, definované jako nejmenší β takové, že existují poruchy $\Delta\mathbf{A}$ a Δb splňující $\|\Delta\mathbf{A}\| \leq \beta\|\mathbf{A}\|$, $\|\Delta b\| \leq \beta\|b\|$ a x_k je řešením porušené soustavy, $(\mathbf{A} + \Delta\mathbf{A})x_k = b + \Delta b$. V [48] bylo ukázáno, že platí

$$\beta(x_k) = \frac{\|r_k\|}{\|b\| + \|\mathbf{A}\|\|x_k\|}. \tag{1.124}$$

Při praktických výpočtech je norma matice $\mathbf{A}$ nahrazována buď odhadem normy a nebo Frobeniovou normou matice. Číslo $\beta(x_k)$ představuje podle (1.124) lehce vypočitatelnou hodnotu. Na základě její velikosti se můžeme rozhodovat pro zastavení algoritmu iterační metody, jak jsme odůvodnili v kontextu poruch.

Poznamenejme, že platí

$$\mathbf{A}x_k = b - r_k$$

a reziduum r_k představuje poruchu pravé strany Δb, při níž je x_k řešením porušené soustavy $\mathbf{A}x_k = b - \Delta b$. Hodnota $\|r_k\|/\|b\|$ zřejmě reprezentuje relativní velikost poruchy pravé strany. Měříme-li konvergenci touto hodnotou, připouštíme nepřesnosti (poruchy) pouze v pravé straně a nikoliv v matici $\mathbf{A}$. Případy, kdy je matice dána přesně (její prvky nejsou zatíženy žádnými chybami – jsou dány přesnými hodnotami problému reálného světa) a pravá strana nepřesně jsou však velmi řídké. V praxi bývá hodnota $\|r_k\|/\|b\|$ resp. $\|r_k\|/\|r_0\|$ (relativní norma rezidua) velmi často neopodstatněně používána při zastavování algoritmu.

Věříme, že pokud není k dispozici nějaké jiné kritérium, které plyne z přirozenosti úlohy, je normovaná relativní zpětná chyba vhodná pro zastavovací kritétium a měla by být preferována před relativní normou rezidua.

2

O stínovém vektoru

Stínový vektor, nazývaný též levý startovací vektor, je pomocný vektor, který vstupuje do Lanczosova procesu jako parametr. Tento parametr generuje společně s maticí $\mathbf{A}^T$ levý Krylovův prostor, potřebný k určení vektorů Lanczosovy báze. Jeho volba zřejmě ovlivňuje kvalitu počítané Lanczosovy báze a tím i konvergenci metod pro řešení soustavy lineárních rovnic, které tuto bázi používají ke konstrukci aproximace řešení. Dosud však nebylo ukázáno ani vysvětleno, jak závisí kvalita budované Lanczosovy báze na volbě stínového vektoru. Ani my nezodpovíme tuto fundamentální otázku. Navrhneme však různé cesty ke zkoumání tohoto parametru a ukážeme volby stínového vektoru, kterými lze například docílit rovnosti některých reziduí Lanczosovy metody pro řešení systému lineárních rovnic s rezidui jiné krylovovské metody. Budeme diskutovat otázku proč jsou konvergenční křivky klasických krylovovských metod často velmi blízké konvergenční křivce GMRES *vždy, když dochází k jejímu dostatečně rychlému poklesu. Provedeme experiment, ve kterém se pokusíme numericky určit optimální stínový vektor.*

V předchozí kapitole jsme vysvětlili, jak lze efektivně počítat bázové vektory Krylovova prostoru takové, že platí

$$v_{k+1} \perp \mathcal{K}_k(\mathbf{A}^T, \widetilde{r}_0), \tag{2.1}$$

kde $\widetilde{r}_0$ je pomocný vektor. Podmínka (2.1) měla za následek, že se rekurence (1.14), podle které jsme se rozhodli počítat bázové vektory, redukovala na tříkrokovou rekurenci a následující bázový vektor bylo možné vypočíst pouze ze znalosti předchozích dvou bázových vektorů. Rezidua a aproximace řešení soustavy (1.1) vyhovující podmínkám

$$x_k \in x_0 + \mathcal{K}_k(\mathbf{A}, r_0), \qquad r_k \perp \mathcal{K}_k(\mathbf{A}^T, \widetilde{r}_0),$$

jsme vyjádřili pomocí krátkých rekurencí, algoritmem BiCG. Definovali jsme řadu dalších metod (QMR, CGS, BiCGStab atd.) postavených na základní podmínce (2.1) a využívajících výhod Lanczosovy báze. Všechny tyto metody jsou laciné, co se počtu operací na jednu iteraci týče, a kvalita vypočítávaných aproximací řešení je mnohdy srovnatelná s kvalitou aproximací řešení, kterou poskytují metody s dlouhými rekurencemi (GMRES, FOM). Metody používající Lanczosovu bázi nám tak často dávají možnost získat „dobrou“ aproximaci řešení z variety $x_0 + \mathcal{K}_k(\mathbf{A}, r_0)$ při malých nárocích na paměť počítače a s nízkými výpočetními náklady. Nad jejich konvergencí se však vznáší mnoho otazníků. Jsme přesvědčeni, že klíč k zodpovězení mnoha otázek týkajících se konvergence těchto metod je skryt v pochopení úlohy stínového vektoru $\widetilde{r}_0$ v Lanczosově procesu.

Jednou z možností, jak pohlížet na celý problém volby stínového vektoru, je tato. Uvažujme *obecnou tříkrokovou rekurenci* (pro $k = 1, 2, \dots$)

$$\mathbf{t} \;=\; \mathbf{A}v_k - \alpha_k v_k - \beta_{k-1} v_{k-1}, \quad \gamma_k = \|\mathbf{t}\|, \quad v_{k+1} = \mathbf{t}/\gamma_k, \tag{2.2}$$

$v_1 \equiv r_0/\|r_0\|$, $\beta_0 = 0$, $v_0 = \mathbf{o}$. Nesymetrický Lanczosův algoritmus počítá vektory Lanczosovy báze tříkrokovou rekurencí (2.2), přičemž koeficienty α_k a β_{k-1} jsou určeny podmínkou kolmosti bázového vektoru v_{k+1} na levý prostor $\mathcal{K}_k(\mathbf{A}^T, \widetilde{r}_0)$ – rekurenci (2.2) s takto

určenými koeficienty α_k a β_{k-1} budeme nazývat *Lanczosovou rekurencí*. Různou volbou vektoru $\widetilde{r}_0$ budou určeny různé koeficienty α_k a β_{k-1}. Lze tedy tvrdit, že posloupnosti Lanczosových vektorů určených vektory r_0, $\widetilde{r}_0$ a maticí $\mathbf{A}$ jsou podmnožinou posloupností vektorů počítaných rekurencí (2.2), ve které jsou koeficienty α_k a β_{k-1} voleny libovolně. Není však vůbec zřejmé, jak je podmnožina posloupností Lanczosových vektorů „velká". K pochopení významu stínového vektoru v tomto pojetí je potřeba určit vztah mezi Lanczosovou rekurencí a obecnou tříkrokovou rekurencí (2.2). Pro jednoduchost budeme v celé této kapitole předpokládat, že platí

$$\dim(\mathcal{K}_n(\mathbf{A}, r_0)) = n. \tag{2.3}$$

2.1 Stínový vektor a tříkroková rekurence

O vztahu mezi Lanczosovými vektory a vektory počítanými obecnou tříkrokovou rekurencí (2.2) vypovídá do značné míry věta od Anne Greenbaum [25].

Věta 2.1 (*„**Greenbaum**"*). *Provedeme-li ne více než* $(n+2)/2$ *kroků podle předpisu* (2.2), *kde* α_k *a* β_{k-1} *mohou být téměř libovolné, existuje vektor* $\widetilde{r}_0$ *takový, že vektory a koeficienty v* (2.2) *jsou stejné jako příslušné vektory a koeficienty počítané* NL *algoritmem.*

Věta 2.1 říká, že můžeme téměř libovolně předepsat koeficienty α_k a β_{k-1} obecné tříkrokové rekurence (2.2) a pokud neprovedeme více než $(n+2)/2$ kroků, lze tuto rekurenci chápat jako Lanczosovou rekurenci. Pojmem „téměř libovolně" je v [25] myšleno, že míra množiny koeficientů, pro které uvedené tvrzení neplatí, je nulová. Příkladem rekurence (2.2), kterou nelze chápat jako Lanczosovu, je rekurence, v níž jsou nulové všechny koeficienty α_k a β_{k-1}. Potom (2.2) reprezentuje mocninnou metodu a při pokusu počítat tyto vektory NL algoritmem skončíme v kroku 2, kdy nelze vypočíst ani jeden z koeficientů. V následujícím textu se pokusíme o zobecnění Greenbaumové věty a o přesnou specifikaci jejích předpokladů.

Při zkoumání stínového vektoru bylo v [25] využito vlastnosti skalárního součinu

$$(u, \mathbf{A}^T v) = (\mathbf{A}u, v), \tag{2.4}$$

kde u a v jsou libovolné vektory. Totéž platí, vyskytuje-li se v (2.4) místo samotné matice $\mathbf{A}$ její mocnina. Aplikujeme-li (2.4) na ortogonální podmínky, jež splňuje vektor v_{k+1}, dostáváme

$$0 = (v_{k+1}, (\mathbf{A}^T)^i \widetilde{r}_0) = (\mathbf{A}^i v_{k+1}, \widetilde{r}_0), \qquad i = 0, \ldots, k-1$$

a tudíž

$$v_{k+1} \perp \mathcal{K}_k(\mathbf{A}^T, \widetilde{r}_0) \Leftrightarrow \widetilde{r}_0 \perp \mathcal{K}_k(\mathbf{A}, v_{k+1}), \quad k = 1, 2, \ldots. \tag{2.5}$$

Definujeme-li

$$\mathcal{W}_k(\mathbf{A}, v_2, \ldots, v_{k+1}) \equiv \operatorname{span} \begin{pmatrix} v_2, & v_3, & v_4, & v_5 & \ldots & v_{k+1}, \\ & \mathbf{A}v_3, & \mathbf{A}v_4, & \mathbf{A}v_5, & \ldots & \mathbf{A}v_{k+1}, \\ & & \mathbf{A}^2 v_4, & \mathbf{A}^2 v_5, & \ldots & \mathbf{A}^2 v_{k+1}, \\ & & & \ddots & & \vdots \\ & & & & \ddots & \vdots \\ & & & & & \mathbf{A}^{k-1} v_{k+1} \end{pmatrix},$$

t.j

$$\mathcal{W}_k(\mathbf{A}, v_2, \ldots, v_{k+1}) \equiv \bigcup_{i=1}^{k} \mathcal{K}_i(\mathbf{A}, v_{i+1}), \tag{2.6}$$

platí podle (2.5)

$$\widetilde{r}_0 \perp \mathcal{W}_k(\mathbf{A}, v_2, \ldots, v_{k+1}). \tag{2.7}$$

Jsou-li vektory v_k počítány rekurencí (2.2) a volíme-li vektor $\widetilde{r}_0$ podle (2.7), plyne z jednoznačnosti určení Lanczosových vektorů, že NL algoritmus s parametry r_0 a $\widetilde{r}_0$ počítá přesně stejné vektory a koeficienty jako rekurence (2.2), pokud ovšem nedojde k předčasnému ukončení algoritmu. Pokud je $k > n/2$, je obecně $\dim \mathcal{W}_k(\mathbf{A}, v_2, \ldots, v_{k+1}) = n$ a tudíž není možné zvolit $\widetilde{r}_0 \neq \mathbf{o}$ splňující (2.7). To vše bylo řečeno v článku [25].

V následující větě rozšíříme tvrzení Anne Greenbaum a budeme přesně specifikovat předpoklady, za kterých toto tvrzení platí. Pro zjednodušení zápisu budeme psát v dalším $\mathcal{W}_k$ místo $\mathcal{W}_k(\mathbf{A}, v_2, \ldots, v_{k+1})$.

Věta 2.2 *Vektory $v_1, \ldots, v_{k+1}$ vypočtené tříkrokovou rekurencí* (2.2) *lze počítat nesymetrickým Lanczosovým algoritmem* 1.2 *právě tehdy, pokud platí $r_0 \notin \mathcal{W}_k$ a*

$$\beta_j \neq 0, \qquad j = 1, \ldots, k-1. \tag{2.8}$$

Je-li $k \leq n/2$, lze vektory určené tříkrokovou rekurencí (2.2) *s koeficinty splňujícími podmínku* (2.8) *vždy považovat za Lanczosovy vektory vypočtené algoritmem* 1.2 (NL).

Důkaz. Předpokládejme nejdříve, že vektory $v_1, \ldots, v_{k+1}$ jsou vypočteny tříkrokovou rekurencí (2.2) s koeficienty splňujícími (2.8) a že platí $r_0 \notin \mathcal{W}_k$. Ukážeme, že lze tyto vektory rovněž vypočíst algoritmem 1.2 (NL).

Protože $r_0 \notin \mathcal{W}_k$, lze nalézt vektor $\widetilde{r}_0$ takový, že platí $\widetilde{r}_0^T r_0 \neq 0$ a $\widetilde{r}_0 \perp \mathcal{W}_k$. Definujme vektor w_1 vztahem $w_1 \equiv \widetilde{r}_0 / \widetilde{r}_0^T v_1$, t.j. $w_1^T v_1 = 1$. Z nenulovosti koeficientů β_j plyne, že lze definovat vektory $w_2, \ldots, w_k$ rekurentním vztahem

$$w_{j+1} \equiv \beta_j^{-1}(\mathbf{A}^T w_j - \alpha_j w_j - \gamma_{j-1} w_{j-1}), \qquad j = 1, \ldots, k-1, \tag{2.9}$$

přičemž pro $j = 1$ definujeme $\gamma_0 \equiv 0$, $w_0 \equiv \mathbf{o}$. Nyní postačí ukázat, že je možné koeficienty β_j $(j = 1, \ldots, k-1)$ a α_j $(j = 1, \ldots, k)$ vyjádřit ve tvaru

$$\alpha_j = w_j^T \mathbf{A} v_j, \qquad \beta_j = v_{j+1}^T \mathbf{A}^T w_j. \tag{2.10}$$

Potom budou zřejmě vztahy (2.2), (2.9) a (2.10) totožné s algoritmem NL, t.j. vektory $v_2, \ldots, v_{k+1}$ lze počítat algoritmem NL.

Protože je vektor $\widetilde{r}_0$ kolmý na prostor $\mathcal{W}_k$, plyne z (2.5)

$$v_{j+1} \perp \mathcal{K}_j(\mathbf{A}^T, \widetilde{r}_0), \quad \text{t.j.} \quad v_{j+1}^T w_i = 0, \qquad i \leq j, \quad j = 1, \ldots, k. \tag{2.11}$$

Přenásobením rekurence (2.9) vektorem v_{j+1}^T a využitím (2.11) dostáváme

$$v_{j+1}^T w_{j+1} \beta_j = v_{j+1}^T \mathbf{A}^T w_j. \tag{2.12}$$

Ukažme sporem, že $v_{j+1}^T \mathbf{A}^T w_j \neq 0$, $j = 1, \ldots, k-1$. Nechť tedy platí $v_{j+1}^T \mathbf{A}^T w_j = 0$. Potom zřejmě $v_{j+1}^T (\mathbf{A}^T)^j \widetilde{r}_0 = 0$ a vektor $\widetilde{r}_0$ je kolmý na prostor $\mathcal{W}_k \cup \mathbf{A}^j v_{j+1}$.

Indukcí ukážeme, že $\mathbf{A}^j v_{j+1} \notin \mathcal{W}_k$ pro $j = 1, 2, \ldots, k-1$. Podle předpokladu platí $\mathbf{A}^0 v_1 \notin \mathcal{W}_k$. Přenásobíme-li vztah pro vektor v_{j+1} maticí $\mathbf{A}^{j-2}$, $(j \geq 2)$, obdržíme rekurenci

$$\mathbf{A}^{j-2} v_{j+1} = \gamma_j^{-1}(\mathbf{A}^{j-1} v_j - \alpha_j \mathbf{A}^{j-2} v_j - \beta_{j-1} \mathbf{A}^{j-2} v_{j-1}).$$

Protože je $\mathbf{A}^{j-2} v_{j+1} \in \mathcal{W}_k$, $\mathbf{A}^{j-2} v_j \in \mathcal{W}_k$, $\beta_{j-1} \neq 0$, $\gamma_j \neq 0$ a podle indukčního předpokladu $\mathbf{A}^{j-2} v_{j-1} \notin \mathcal{W}_k$, platí rovněž $\mathbf{A}^{j-1} v_j \notin \mathcal{W}_k$.

Z rovnosti prostorů

$$\operatorname{span}\{v_1, v_2, \ldots, v_k, v_{k+1}, \mathbf{A} v_{k+1}, \ldots \mathbf{A}^{k-1} v_{k+1}\} = \mathcal{K}_{2k}(\mathbf{A}, r_0)$$

a z $\mathcal{W}_k \subseteq \mathcal{K}_{2k}(\mathbf{A}, r_0)$, $v_1 \notin \mathcal{W}_k$, plyne

$$\mathcal{W}_k \cup r_0 = \mathcal{K}_{2k}(\mathbf{A}, r_0).$$

Jelikož $\mathbf{A}^j v_{j+1} \notin \mathcal{W}_k$, $\mathbf{A}^j v_{j+1} \in \mathcal{K}_{2k}(\mathbf{A}, r_0)$, je rovněž

$$\mathcal{W}_k \cup \mathbf{A}^j v_{j+1} = \mathcal{K}_{2k}(\mathbf{A}, r_0), \qquad j = 1, \ldots, k-1.$$

Je-li vektor $\widetilde{r}_0$ kolmý na $\mathcal{W}_k \cup \mathbf{A}^j v_{j+1}$, je kolmý i na vektor r_0, což je spor s předpoklady.

V předchozím jsme ukázali, že je $v_{j+1}^T \mathbf{A}^T w_j \neq 0$. Podle (2.12) je rovněž $v_{j+1}^T w_{j+1} \neq 0$ a koeficient β_j lze vyjádřit ve tvaru

$$\beta_j = \frac{v_{j+1}^T \mathbf{A}^T w_j}{v_{j+1}^T w_{j+1}}, \qquad j = 1, \ldots, k-1.$$

Přenásobíme-li rekurenci

$$v_{j+2} = \gamma_{j+1}^{-1}(\mathbf{A} v_{j+1} - \alpha_{j+1} v_{j+1} - \beta_j v_j).$$

vektorem w_j^T, obdržíme s využitím (2.11) a $v_j^T w_j \neq 0$ koeficient β_j ve tvaru

$$\beta_j = \frac{v_{j+1}^T \mathbf{A}^T w_j}{v_j^T w_j}.$$

Nutně tedy platí $v_{j+1}^T w_{j+1} = v_j^T w_j = \cdots = v_1^T w_1 = 1$ a

$$\beta_j = v_{j+1}^T \mathbf{A}^T w_j, \qquad j = 1, \ldots, k-1.$$

Tvar koeficientu α_j $(j = 1, \ldots, k)$ lehce obdržíme z rekurence pro vektor v_{j+1}, kterou přenásobíme vektorem w_j a využijeme přitom (2.11) a $v_j^T w_j = 1$.

Ukázali jsme, že předpoklady $r_0 \notin \mathcal{W}_k$ a $\beta_j \neq 0$, $j = 1, \ldots, k$ jsou postačující pro to, aby bylo možné počítat vektory $v_1, v_2, \ldots, v_{k+1}$ Lanczosovou rekurencí. Nyní vysvětlíme, že jsou tyto předpoklady nutné.

Pokud je $r_0 \in \mathcal{W}_k$, není zřejmě možné nalézt stínový vektor takový, který by byl kolmý na $\mathcal{W}_k$ a zároveň nebyl kolmý na vektor r_0. Je-li některý z koeficientů β_j nulový, dojde k ukončení NL při výpočtu vektoru w_{j+1}.

Věnujme se nyní druhé části věty, která je v podstatě větou 2.1 s přesnou specifikací předpokladů, t.j. že všechny β_j musí být nenulové. Podle první části věty 2.2 stačí ukázat, že je-li $k \leq n/2$ platí $r_0 \notin \mathcal{W}_k$.

Protože je $k \leq n/2$, jsou vektory $v_2, \ldots, v_{k+1}, \mathbf{A} v_{k+1}, \mathbf{A}^{k-1} v_{k+1}$, kterých je $2k-1 < n$, lineárně nezávislé (předpokládáme, že platí (2.3)). Ukážeme, že všechny ostatní vektory

z $\mathcal{W}_k$ leží v jejich lineárním obalu. Přenásobíme-li rekurenci (2.2) pro $j = 3, \ldots, k$ maticí $\mathbf{A}^{i-1}$, kde i je libovolné celé číslo, a vyjádříme-li $\mathbf{A}^i v_j$, zjistíme, že

$$\mathbf{A}^i v_j \in \operatorname{span}\{\mathbf{A}^{i-1} v_{j+1}, \mathbf{A}^{i-1} v_j, \mathbf{A}^{i-1} v_{j-1}\}. \tag{2.13}$$

Z (2.13) plyne $\mathbf{A}v_j \in \operatorname{span}\{v_{j+1}, v_j, v_{j-1}\}$ a tedy $\mathbf{A}v_j \in \mathcal{W}_k$ pro $j = 3, \ldots, k$, $\mathbf{A}^2 v_j \in \operatorname{span}\{\mathbf{A}v_{j+1}, \mathbf{A}v_j, \mathbf{A}v_{j-1}\}$ a proto $\mathbf{A}^2 v_j \in \mathcal{W}_k$ pro $j = 4, \ldots, k$ a indukcí plyne $\mathbf{A}^i v_j \in \mathcal{W}_k$ $j = i + 2, \ldots, k$, $i = 1, \ldots, k - 2$.

Dimenze prostoru $\mathcal{W}_k$ je tedy rovna $2k - 1$ a vektor $r_0 \notin \mathcal{W}_k$, jinak by v prostoru $\mathcal{W}_k$ leželo $2k$ lineárně nezávislých vektorů $v_1, v_2, \ldots, v_{k+1}, \mathbf{A}v_{k+1}, \mathbf{A}^{k-1} v_{k+1}$. □

Věty 2.1 a 2.2 vysvětlují, že každou tříkrokovou rekurenci (2.2) s nenulovými koeficienty β_j lze považovat za rekurenci počítající Lanczosovy vektory až do kroku $k \leq n/2$. Pro vyšší hodnoty k počítá rekurence (2.2) Lanczosovy vektory pouze pokud počáteční reziduum neleží v prostoru $\mathcal{W}_k(\mathbf{A}, v_2, \ldots, v_{k+1})$ definovaném v (2.6).

2.2 Stínový vektor a tříkroková krylovovská metoda

Vektory Lanczosovy báze v_{k+1} počítané algoritmem 1.2 (NL) lze vhodně přenásobit skalárem tak, aby ležely ve varietě $r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$. Stačí položit

$$\gamma_k = -(\alpha_k + \beta_{k-1}). \tag{2.14}$$

Potom lze Lanczosovy vektory považovat za rezidua, označme je $r_k \equiv v_{k+1}$. K rekurencím algoritmu 1.2 přidejme navíc rekurenci pro počítání příslušných aproximací řešení. Tuto rekurenci lze lehce odvodit z rekurence pro počítání Lanczosových vektorů, uvážíme-li, že $r_k = b - \mathbf{A}x_k$. Výsledný algoritmus 2.1 označme LMA (*Lanczos method's algorithm*).

ALGORITMUS 2.1. *Algoritmus Lanczosovy metody* (LMA)

input x_0, $\mathbf{A}$, b, $\widetilde{r}_0$ **for** $k = 1, \ldots$

initialization

$r_{-1} = \mathbf{o}$
$x_{-1} = \mathbf{o}$
$w_0 = \mathbf{o}$
$\beta_0 = 0$
$\gamma_0 = 0$
$w_1 = \widetilde{r}_0 / \widetilde{r}_0^T r_0$

$\alpha_k = w_k^T \mathbf{A} r_{k-1}$
$\gamma_k = -(\alpha_k + \beta_{k-1})$
$r_k = \gamma_k^{-1}(\mathbf{A}r_{k-1} - \alpha_k r_{k-1} - \beta_{k-1} r_{k-2})$
$x_k = \gamma_k^{-1}(-r_{k-1} + \alpha_k x_{k-1} + \beta_{k-1} x_{k-2})$
$\mathbf{t} = \mathbf{A}^T w_k - \alpha_k w_k - \gamma_{k-1} w_{k-1}$
$\beta_k = r_k^T \mathbf{t}$
$w_{k+1} = \mathbf{t}/\beta_k$

end for

Algoritmus 2.1 lze považovat za jeden z možných algoritmů, který počítá rezidua a aproximace řešení Lanczosovy metody pro řešení systému lineárních rovnic, platí

$$x_k \in x_0 + \mathcal{K}_k(\mathbf{A}, r_0), \quad r_k \perp \mathcal{K}_k(\mathbf{A}^T, \widetilde{r}_0). \tag{2.15}$$

Vektory r_k a x_k vypočtené algoritmem 2.1 jsou ideálně (v přesné aritmetice) totožné s rezidui a aproximacemi řešení určenými algoritmem 1.6 (BiCG).

Uvažujme nyní obecnou tříkrokovou rekurenci

$$r_k = \gamma_k^{-1}(\mathbf{A}r_{k-1} - \alpha_k r_{k-1} - \beta_{k-1} r_{k-2}), \tag{2.16}$$

$$x_k = \gamma_k^{-1}(-r_{k-1} + \alpha_k x_{k-1} + \beta_{k-1} x_{k-2}), \tag{2.17}$$

kde α_k a β_{k-1} jsou libovolné koeficienty takové, že $\alpha_k + \beta_{k-1} \neq 0$ a koeficient γ_k nechť je volen podle (2.14). Potom platí

$$r_k \in r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$$

a rekurencemi (2.16) lze reprezentovat libovolnou krylovovskou tříkrokovou metodu. Protože r_k jsou pouze speciálně normované bázové vektory v_{k+1}, můžeme použít výsledků z předchozí sekce. Platí následující věta, jež je přímým důsledkem věty 2.2.

Věta 2.3 *Rezidua a aproximace řešení libovolné krylovovské tříkrokové metody s nenulovými koeficienty β_j můžeme až do kroku $k \leq n/2$ chápat jako rezidua a aproximace řešení* LM *s vhodně zvoleným stínovým vektorem a počítat je lze algoritmem* 2.1. *Totéž platí, pokud je $k > n/2$ a $r_0 \notin \mathcal{W}_k(\mathbf{A}, r_1, \ldots, r_k)$.*

Zřejmě je možné volit koeficienty v tříkrokové rekurenci (2.2) téměř libovolně a přesto lze tuto rekurenci chápat jako rekurenci algoritmu Lanczosovy metody. V prvních $n/2$ krocích aproximace řešení Lanczosovy metody vůbec nemusí konvergovat k řešení soustavy (1.1).

Z druhé strany nám věta 2.3 v podstatě dává možnost převést problém volby stínového vektoru v Lanczosově metodě na problém nalezení vhodné krylovovské tříkrokové metody. Chceme-li například, aby rezidua Lanczosovy metody byla blízká reziduím jiné krylovovské metody v prvních $n/2$ krocích, stačí nalézt tříkrokovou krylovovskou metodu, jejíž rezidua splňují požadované podmínky. Ze znalosti reziduí této tříkrokové krylovovské metody již jsme schopni podle (2.7) určit vhodný stínový vektor.

Poznamenejme, že roste-li dimenze Krylovových prostorů $\mathcal{K}_k(\mathbf{A}, r_0)$ a $\mathcal{K}_k(\mathbf{A}^T, \widetilde{r}_0)$, je v n-tém kroku $r_n \perp \mathcal{K}_n(\mathbf{A}^T, \widetilde{r}_0)$ a $w_{n+1} \perp \mathcal{K}_n(\mathbf{A}, r_0)$ a tudíž $r_n = \mathbf{o} = w_{n+1}$. Podle (1.39), (1.40), (1.41) a (1.45) potom platí

$$\mathbf{AV} = \mathbf{VT}, \quad \mathbf{A}^T\mathbf{W} = \mathbf{W}\mathbf{T}^T, \quad \mathbf{V}^T\mathbf{W} = \mathbf{I}, \tag{2.18}$$

kde $\mathbf{V} \in \mathbb{R}^{n\times n}$ je matice se sloupci $r_0, \ldots, r_{n-1}$, $\mathbf{W} \in \mathbb{R}^{n\times n}$ se sloupci $w_1, \ldots, w_n$ a $\mathbf{T} \in \mathbb{R}^{n\times n}$ je třídiagonální matice složená z koeficientů α_k, β_{k-1}, γ_k.

Uvedený postup lze obrátit v následujícím smyslu. Předpokládejme, že je dána krylovovská tříkroková metoda s nenulovými koeficienty β_j taková, že platí $r_{n-1} \neq \mathbf{o}$, $r_n = \mathbf{o}$. Zapíšeme-li tříkrokovou rekurenci (2.16) v kroku n maticově, platí

$$\mathbf{AV} = \mathbf{VT}.$$

Definujme nyní $\mathbf{W} \equiv \mathbf{V}^{-T}$. Matice $\mathbf{W}$, $\mathbf{V}$ a $\mathbf{T}$ potom zřejmě splňují (2.18) a určíme-li stínový vektor $\widetilde{r}_0$ jako první sloupec matice $\mathbf{V}^{-T}$, počítá zřejmě algoritmus 2.1 se startovacím vektorem $\widetilde{r}_0$ přesně stejné vektory a aproximace řešení jako uvažovaná tříkroková rekurence (to plyne z jednoznačnosti určení lanzcosových vektorů).

Algoritmus 1.2 (NL) nebo 2.1 (LMA) lze podle (2.18) chápat jako algoritmy na transformaci matice $\mathbf{A}$ na třídiagonální tvar. Matice $\mathbf{T}$ je zřejmě podobná matici $\mathbf{A}$. Z předchozího postupu rovněž plyne, že pomocí stínového vektoru $\widetilde{r}_0$ lze parametrizovat všechny třídiagonální matice $\mathbf{T}$ podobné matici $\mathbf{A}$, s vyjímkou těch matic $\mathbf{T}$, které mají některý prvek vedlejších diagonál nulový. Shrňme vše v následující větě.

Věta 2.4 *Rezidua a aproximace řešení libovolné krylovovské tříkrokové metody s nenulovými koeficienty β_j pro kterou je $r_{n-1} \neq \mathbf{o}$, $r_n = \mathbf{o}$ lze chápat jako rezidua a aproximace řešení* LM *s vhodně zvoleným stínovým vektorem a počítat je algoritmem* 2.1 (LMA). *Každou třídiagonální matici $\mathbf{T}$ která je podobná matici $\mathbf{A}$ a v jejíchž vedlejších diagonálách se nevyskytuje nulový prvek lze chápat jako výsledek výpočtu algoritmu* 1.2 (NL) *s vhodně zvoleným startovacím parametrem $\widetilde{r}_0$.*

2.3 Lanczosova metoda a rezidua jiné krylovovské metody

Myšlenku vyjádřenou vztahem (2.5) lze použít následujícím způsobem. Předpokládejme, že r_k je libovolné reziduum,

$$r_k \in r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0). \tag{2.19}$$

Volíme-li stínový vektor $\widetilde{r}_0$ tak, že splňuje podmínky

$$\widetilde{r}_0 \perp \mathcal{K}_k(\mathbf{A}, r_k), \qquad \widetilde{r}_0^T r_0 \neq 0, \tag{2.20}$$

platí podle (2.5)

$$r_k \perp \mathcal{K}_k(\mathbf{A}^T, \widetilde{r}_0). \tag{2.21}$$

Pokud nyní vypočteme k-té reziduum Lanczosovy metody se startovacím parametrem $\widetilde{r}_0$ voleným podle (2.20), splňuje toto reziduum, označme ho r_k^L, podmínky (2.19) i (2.21). Jelikož jsou rezidua r_k^L i r_k určena těmito podmínkami jednoznačně, nutně platí

$$r_k = r_k^L. \tag{2.22}$$

Právě uvedený postup naznačuje, že pomocí algoritmů s krátkými rekurencemi, které realizují Lanczosovu metodu pro řešení soustavy lineárních rovnic, půjde pravděpodobně téměř vždy (pro skoro každý vektor r_0 a skoro každou matici $\mathbf{A}$) vypočíst skoro každý reziduový vektor z variety $r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$, tedy i např. i reziduum metody GMRES nebo FOM. Pro korektní důkaz tohoto tvrzení bychom však museli ukázat, že pro skoro každý r_0 a matici $\mathbf{A}$ a $r_k \in r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$ existuje stínový vektor takový, že splňuje podmínky (2.20) a zároveň že nedojde k předčasnému ukončení algoritmu realizujícího Lanczosovu metodu. Pohovořme v krátkosti o tomto problému (inspirací nám je práce [34]).

Uvažujme jen takové vektory r_k, které lze vyjádřit ve tvaru $r_k = \varphi_k(\mathbf{A})r_0$, kde $\varphi_k(\xi)$ je polynom stupně k, $\varphi_k(0) = 1$ (míra množiny všech reziduí z variety $r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$ které nelze takto vyjádřit je nulová). Je-li $k < n/2$, platí

$$r_0 \notin \mathcal{K}_k(\mathbf{A}, r_k) \tag{2.23}$$

(jinak by byl vektor $\mathbf{A}^{2k-1}r_0$ netriviální lineární kombinací vektorů $r_0, \ldots, \mathbf{A}^{2k-2}r_0$ což je spor s předpokladem $\dim(\mathcal{K}_n) = n$). Nutně tedy skoro každý vektor $\widetilde{r}_0$ z prostoru

$$\mathcal{H}_k \equiv \mathcal{K}_k(\mathbf{A}, r_k)^\perp$$

splňuje podmínky (2.20) a lze ho vyjádřit ve tvaru

$$\widetilde{r}_0 = \sum_{i=1}^{n-k} \xi_i h_i,$$

kde h_i tvoří bázi prostoru $\mathcal{H}_k$. Koeficienty γ_k a β_k, kterými v algoritmu 2.1 (LMA) dělíme, lze potom považovat za racionální funkce proměnných $\xi_1, \ldots, \xi_{n-k}$. Tyto racionální funkce jsou buď netriviální a tedy skoro-všude nenulové (pro skoro každý vektor $\widetilde{r}_0 \in \mathcal{H}_k$ nedojde k předčasnému ukončení algoritmu) a nebo je některá racionální funkce identicky rovna nule a pak dojde k předčasnému ukončení algoritmu 2.1 (LMA) pro každý vektor $\widetilde{r}_0 \in \mathcal{H}_k$.

Pro lepší představu uveďme příklad. Protože $r_0 \notin \mathcal{K}_k(\mathbf{A}, r_k)$, je skalární součin

$$(\widetilde{r}_0, r_0) = \sum_{i=1}^{n-k} \xi_i (h_i, r_0),$$

kterým dělíme při výpočtu vektoru w_1, netriviální lineární funkcí proměnných ξ_i. Dále

$$\alpha_1 = -\gamma_1 = (w_1, \mathbf{A}r_0) = \frac{\sum_{i=1}^{n-k} \xi_i (h_i, \mathbf{A}r_0)}{\sum_{i=1}^{n-k} \xi_i (h_i, r_0)},$$

je triviální funkcí pouze tehdy, pokud platí podmínka $\mathbf{A}r_0 \in \mathcal{K}_k(\mathbf{A}, r_k)$. Tato podmínka je však zřejmě splněna jen za velmi speciálních okolností (vyhovuje-li matice $\mathbf{A}$, vektor r_0 a vektor r_k určitým rovnostem). Lze tedy tvrdit, že koeficient γ_1 (jakožto racionální funkce proměnných ξ_i) je pro skoro všechny vektory r_0, matice $\mathbf{A}$ a vektory $r_k \in r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$ netriviální.

Další koeficienty γ_k a β_k lze vyjádřit jako již více komplikované racionální funkce, které jsou identicky rovny nule jen za velmi speciálních podmínek (pouze pro speciálně volená rezidua r_k z variety $r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$ splňující určité rovnice nebo pro speciálně volené počáteční reziduum a matici $\mathbf{A}$). Proto lze očekávat, že pro $k < n/2$ lze pomocí Lanczosova algoritmu 2.1 (LMA) vypočíst téměř vždy (pro skoro všechny vektory r_0 a matice $\mathbf{A}$) skoro všechna rezidua r_k z variety $r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$. Tentýž výsledek zřejmě platí i pro algoritmus 1.6 (BiCG), pouze se zvětší množina vektorů r_k, které nelze vypočíst z důvodů předčasného ukončení algoritmu. Tato množina reziduí, které nelze vypočíst algoritmem BiCG a lze je vypočíst algoritmem LMA, má zřejmě míru nula.

Pro $k > n/2$ již nemusí být podmínka (2.23) splněna pro každé reziduum z variety $r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$, nicméně lze očekávat, že (2.23) je splněna pro skoro všechna rezidua a výsledky odvozené v předchozím textu lze zobecnit na všechna $k < n$.

Předchozí úvahy dokumentují schopnost algoritmů s krátkými rekurencemi vypočíst v daném kroku stejně „dobrou“ aproximaci řešení jako metody s dlouhými rekurencemi. Avšak cesta, kterou se algoritmy s krátkými rekurencemi k dané optimální aproximaci řešení dopracují zřejmě už optimální není a v konečné aritmetice počítače mohou být některá optimální rezidua (a aproximace řešení) pro algoritmy které realizují Lanczosovu metodu prakticky nedosažitelná (viz. obrázek 2.3).

Myšlenku volit stínový vektor tak, aby Lanczosova metoda vypočetla téměř libovolné reziduum z variety $r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$ lze dále rozvést. Představme si, že jsou dána rezidua libovolné krylovovské metody $r_1, \ldots, r_k$ (v podstatě libovolná posloupnost reziduí, která leží v příslušných varietách). Potom lze zřejmě definovat prostor $\mathcal{W}_k(\mathbf{A}, r_1, \ldots, r_k)$ stejně jako v (2.6),

$$\mathcal{W}_k(\mathbf{A}, r_1, \ldots, r_k) \equiv \operatorname{span} \begin{pmatrix} r_1, & r_2, & r_3, & r_4 & \ldots & r_k, \\ & \mathbf{A}r_2, & \mathbf{A}r_3, & \mathbf{A}r_4, & \ldots & \mathbf{A}r_k, \\ & & \mathbf{A}^2 r_3, & \mathbf{A}^2 r_4, & \ldots & \mathbf{A}^2 r_k, \\ & & & \ddots & & \vdots \\ & & & & \ddots & \vdots \\ & & & & & \mathbf{A}^{k-1} r_k \end{pmatrix}.$$

Z předchozího odstavce víme, že pokud jsou rezidua $r_1, \ldots, r_k$ generována tříkrokovou rekurencí s nenulovými koeficienty β_i, lze volit stínový vektor podle (2.7) tak, že algoritmus Lanczosovy metody 2.1 (LMA) počítá v prvních $(n+2)/2$ krocích přesně stejná rezidua a aproximace jako uvažovaná tříkroková rekurence. Jsou-li však rezidua $r_1, \ldots, r_k$ generována dlouhou rekurencí, je obecně vektor r_0 obsažen v prostoru $\mathcal{W}_k(\mathbf{A}, r_1, \ldots, r_k)$, platí $\mathcal{W}_k(\mathbf{A}, r_1, \ldots, r_k) = \mathcal{K}_{2k}(\mathbf{A}, r_0)$ a volba $\widetilde{r}_0 \perp \mathcal{W}_k(\mathbf{A}, r_1, \ldots, r_k)$ povede k předčasnému ukončení algoritmu LMA v prvním kroku.

V dalším budeme psát pouze $\mathcal{W}_k$ místo $\mathcal{W}_k(\mathbf{A}, r_1, \ldots, r_k)$. V (2.20) jsme volili stínový vektor kolmý na část prostoru $\mathcal{W}_k$ a tím jsme docílili toho, že k-té reziduum Lanczosovy

metody bylo shodné s předepsaným k-tým reziduem z variety $r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$. V tomto duchu lze pokračovat a volit $\widetilde{r}_0$ kolmý na větší podprostor prostoru $\mathcal{W}_k$, který ovšem nesmí obsahovat počáteční reziduum. Takovým podprostorem je například prostor

$$\mathcal{Z}_{2^l} \equiv \bigcup_{i=0}^{l} \mathcal{K}_{2^i}(\mathbf{A}, r_{2^i}), \tag{2.24}$$

kde $2^l \leq k$. Předpokládáme-li, že vedoucí koeficient u polynomu $\varphi_{2^i}(\mathbf{A})r_0 = r_{2^i}$ je nenulový a je-li $k < n/2$, jsou vektory

$$r_1, r_2, \mathbf{A}r_2, r_4, \mathbf{A}r_4, \mathbf{A}^2r_4, \mathbf{A}^3r_4, r_8, \ldots$$

lineárně nezávislé a jejich lineární obal neobsahuje počáteční reziduum (pokud by obsahoval, lehce dostaneme spor s předpokladem (2.3)). Můžeme tedy volit stínový vektor tak, že je kolmý na prostor $\mathcal{Z}_{2^l}$ a zároveň splňuje podmínku $\widetilde{r}_0^T r_0 \neq 0$. Nedojde-li k předčasnému ukončení algoritmu Lanczosovy metody 2.1 (LMA) nastartovaného se vstupním parametrem $\widetilde{r}_0$, počítá tento algoritmus rezidua $r_1^L, \ldots, r_k^L$ taková, že platí

$$r_0^L = r_0,\ r_1^L = r_1,\ r_2^L = r_2,\ r_4^L = r_4,\ r_8^L = r_8 \ldots,\ r_{2^l}^L = r_{2^l}.$$

Je-li například $n = 2^m + 1$, může algoritmus Lanczosovy metody vypočítat až m reziduí jiné krylovovské metody např. GMRES či FOM (zahrnujeme-li do seznamu vypočtených reziduí i r_0). Více reziduí metody s dlouhými rekurencemi zřejmě již pomocí algoritmu LMA obecně vypočíst nejde (pouze ve speciálních případech, kdy jsou v prostoru $\mathcal{Z}_{2^l}$ obsaženy ještě další Krylovovy podprostory $\mathcal{K}_i(\mathbf{A}, r_i)$; tento případ nastává, jen uvažujeme-li speciální tvary dlouhých rekurencí – koeficienty v rekurenci (1.14) musí splňovat určité rovnice).

Jako příklad na demonstraci, kdy jsou v prostoru $\mathcal{Z}_{2^l}$ obsaženy i všechny ostatní Krylovovy podprostory $\mathcal{K}_i(\mathbf{A}, r_i)$, uvádíme následující lemma.

Lemma 2.1 *Předpokládejme, že rezidua r_i krylovovské metody jsou počítána rekurencí*

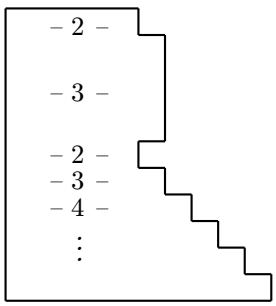
$$r_i = \alpha_i^{(i)}\mathbf{A}r_{i-1} + \sum_{j=0}^{i-1} \alpha_j^{(i)} r_j, \quad \sum_{j=0}^{i-1} \alpha_j^{(i)} = 1, \quad \alpha_i^{(i)} \neq 0. \tag{2.25}$$

Potom vektor r_0 neleží v prostoru $\mathcal{W}_k$ tehdy a jen tehdy, je-li rekurence (2.25) *tvaru*

$$\begin{aligned}
r_1 &= \alpha_1^{(1)}\mathbf{A}r_0 + r_0 \\
r_j &= \alpha_j^{(j)}\mathbf{A}r_{j-1} + \alpha_{j-1}^{(j)} r_{j-1} + \alpha_{j-2}^{(j)} r_{j-2}, \\
&\quad\ \alpha_{j-2}^{(j)} \neq 0,\ j = 2, \ldots, i-1. \\
r_i &= \alpha_i^{(i)}\mathbf{A}r_{i-1} + r_{i-1} \\
r_s &= \alpha_s^{(s)}\mathbf{A}r_{s-1} + \alpha_{s-1}^{(s)} r_{s-1} + \cdots + \alpha_{i-1}^{(s)} r_{i-1}, \\
&\quad\ s = i+1, \ldots, k.
\end{aligned} \tag{2.26}$$

Tvar rekurencí je zobrazen na následujícím schématu.

$r_i =$



Důkaz: Je technického charakteru a uvedli jsme ho v [62, str. 311, Lemma 2]. □

Z lemmatu 2.1 plyne, že je-li $k < n/2$ a jsou-li rezidua počítána rekurencí (2.26) je dimenze prostoru $\mathcal{W}_k$ rovna $2k-1$ a tento prostor neobsahuje počáteční reziduum. Pro $k = 2^l$ je zřejmě $\dim(\mathcal{Z}_{2^l}) = 2k-1 = \dim(\mathcal{W}_{2^l})$. Protože vždy platí $\mathcal{Z}_{2^l} \subseteq \mathcal{W}_{2^l}$, je

$$\mathcal{Z}_{2^l} = \mathcal{W}_{2^l}.$$

V $\mathcal{Z}_{2^l}$ jsou tedy obsaženy všechny Krylovovy podprostory $\mathcal{K}_j(\mathbf{A}, r_j)$, $j \leq 2^l$. Stínový vektor lze v tomto případě sice volit tak, že je kolmý na $\mathcal{Z}_{2^l}$ a platí $\widetilde{r}_0^T r_0 \neq 0$, nicméně, lze ukázat, že pro každý vektor $\widetilde{r}_0$ který splňuje tyto dvě podmínky dojde k předčasnému ukončení algoritmu 2.1 Lanczosovy metody v kroku i (je-li $\alpha_i^{(i-2)} = \beta_i = 0$), viz. [62].

Věříme, že jedna z možných cest zkoumání vztahu mezi metodami s krátkými rekurencemi, reprezentovanými Lanczosovou metodou, a metodami s dlouhými rekurencemi (např. FOM či GMRES) vede skrze zkoumání polohy stínového vektoru $\widetilde{r}_0$ a jednotlivých podprostorů $\mathcal{K}_i(\mathbf{A}, r_i)$ prostoru $\mathcal{W}_k$ generovaného maticí $\mathbf{A}$ a rezidui metody s dlouhými rekurencemi. Snahou je určit, jakým způsobem ovlivňuje odchylka $\widetilde{r}_0$ od prostoru $\mathcal{K}_i(\mathbf{A}, r_i)$ odchylku rezidua r_i^L Lanczosovy metody od daného rezidua r_i (víme pouze, že je-li $\widetilde{r}_0 \perp \mathcal{K}_i(\mathbf{A}, r_i)$ a nedojde-li k předčasnému ukončení algoritmu realizujícího Lanczosovu metodu, je $r_i^L = r_i$). Budeme-li schopni tuto otázku zodpovědět, bude již jen technickým problémem nalézt vhodný stínový vektor takový, že rezidua Lanczosovy metody budou blízká (ve smyslu malé odchylky) reziduím jiné krylovovské metody.

2.4 Jaký stínový vektor je optimální?

V tomto odstavci shrneme výsledky týkající se vztahu metod s krátkými a dlouhými rekurencemi. Pokusíme se z různých pohledů nahlédnout na problematiku optimálního stínového vektoru a naznačit možné cesty k jeho nalezení.

V předchozích sekcích jsme vysvětlili, že v prvních $n/2$ iteracích mohou normy reziduí Lanczosovy metody nabývat téměř libovolných hodnot. Dalším krokem na cestě k pochopení vlivu stínového vektoru na vztah konvergenční křivky Lanczosovy metody resp. QMR a zvolené krylovovské metody (GMRES, FOM) je zodpovězení otázky *„který stínový vektor považovat za optimální a jak tento vektor určit“*. Jde nám o určení hranice, pod kterou se s krátkými rekurencemi již nelze dostat a především potom o zjištění, na čem tato hranice závisí. Rozluštění otázky optimálního stínového vektoru by rovněž mohlo dát dobrý návod k pochopení velmi důležitého případu, kdy je stínový vektor volen jako náhodný vektor z jednotkové koule (uvažujeme rovnoměrné rozdělení) a k případnému určení pravděpodobností se kterými se reziduum Lanczosovy metody bude hodně lišit od rezidua dané krylovovské metody.

Faber a Manteuffel [10] ukázali, že pokud „optimální“ znamená mající minimální normu chyby, přičemž uvažovali normu generovanou skalárním součinem nezávislým na počátečním přiblížení, nelze obecně optimální aproximace řešení počítat krátkými rekurencemi. Ukázali speciální třídu matic, pro které optimální krátké rekurence existují.

Z [10] plyne, že pro většinu nesymetrických systémů nebude možné počítat pomocí rekurencí algoritmu Lanczosovy metody postupně aproximace řešení optimální v nějaké normě nezávislé na počátečním přiblížení. Otevřenou otázkou ovšem zůstává, zda nelze pomocí algoritmu Lanczosovy metody generovat aproximace řešení, které jsou skoro-optimální, tedy velmi blízké optimálním aproximacím řešení. Tuto otázku formulovala Anne Greenbaum ve své práci [26], kde se zabývala vztahem krylovovské metody používající dvoukrokové rekurence, která nalezne řešení soustavy lineárních rovnic v n-krocích a metody GMRES. Popišme stručně, jakého výsledku bylo v [26] dosaženo.

Pro jednoduchost předpokládejme, že matice $\mathbf{A}$ má všechna vlastní čísla $\lambda_1, \ldots, \lambda_n$ různá t.j. je diagonalizovatelná a minimální polynom je roven charakteristickému. Uvažujme dvoukrokovou rekurenci

$$r_k = r_{k-1} - a_k^{-1}\mathbf{A}r_{k-1}, \qquad x_k = x_{k-1} + a_k^{-1}r_{k-1}, \tag{2.27}$$

pro $k = 1, 2, \ldots, n$ a nechť $r_n = \mathbf{o}$, $r_{n-1} \neq \mathbf{o}$. Reziduum r_k lze zřejmě vyjádřit ve tvaru

$$r_k = \varphi_k(\mathbf{A})r_0,$$

kde $\varphi_k(\xi)$ je polynom stupně k, $\varphi_k(0) = 1$ a polynomy $\varphi_k(\xi)$ lze podle (2.27) počítat rekurencí

$$\varphi_k(\xi) = (1 - a_k^{-1}\xi)\varphi_{k-1}(\xi). \tag{2.28}$$

Rekurence (2.27) definuje podobnostní transformaci matice $\mathbf{A}$ na bidiagonální tvar a $\varphi_n(\xi)$ je charakteristický polynom matice $\mathbf{A}$, platí

$$0 = \varphi_n(\xi) = (1 - a_n^{-1}\xi)\varphi_{n-1}(\xi) = (1 - a_n^{-1}\xi)\ldots(1 - a_1^{-1}\xi)$$

a nutně jediná možná volba koeficientů a_k je $a_k = \lambda_k$. Přitom můžeme uvažovat libovolnou permutaci vlastních čísel matice $\mathbf{A}$. V [26] byla zvolena vhodná permutace vlastních čísel následujícím způsobem: za a_n bylo zvoleno takové vlastní číslo, aby byla $\|\varphi_{n-1}(\mathbf{A})r_0\|$ minimální. Podobně, za a_{n-1} bylo zvoleno takové vlastní číslo ze zbývající množiny nepoužitých vlastních čísel, aby bylo $\|\varphi_{n-2}(\mathbf{A})r_0\|$ minimální atd. Pro takto speciálně zvolenou dvoukrokovou krylovovskou metodu bylo v [26] ukázáno, že platí

$$\|r_k\| \leq \sqrt{(k+1)(n-k)}\,\kappa(\mathbf{Z})\,\|r_k^G\|, \tag{2.29}$$

kde $\mathbf{Z}$ je matice vlastních vektorů a r_k^G je k-té reziduum metody GMRES. Poznamenejme, že je-li matice $\mathbf{A}$ normální, je $\kappa(\mathbf{Z}) = 1$. Tolik k práci [26].

Uvažujeme-li tříkrokovou rekurenci (2.16) takovou, že $r_{n-1} \neq \mathbf{o}$, $r_n = \mathbf{o}$ (tu lze téměř vždy chápat jako rekurenci algoritmu LMA Lanczosovy metody viz. věta 2.4), máme zřejmě daleko více možností jak volit koeficienty této rekurence . Lze tedy očekávat, že existuje stínový vektor takový, že rezidua Lanczosovy metody splňují nerovnost (2.29). Bohužel toto tvrzení neumíme přímo dokázat, neboť rezidua speciální dvoukrokové rekurence zvolené v [26] nelze počítat algoritmem Lanczosovy metody LMA (všechny koeficienty β_i jsou rovny nule).

Další prací, ve které je formulován pro nás důležitý výsledek je [40] (Nachtigal).

Věta 2.5 (*„**Nachtigal**“*). *Mezi normami k-tého rezidua metody* GMRES r_k^G *a k-tého rezidua metody* QMR r_k^Q *platí nerovnost*

$$\|r_k^Q\| \leq \kappa(\mathbf{V}_{k+1})\|r_k^G\|, \tag{2.30}$$

kde $\mathbf{V}_{k+1}$ je matice bázových vektorů prostoru $\mathcal{K}_{k+1}(\mathbf{A}, r_0)$ konstruovaných Lanczosovým algoritmem a $\kappa(\cdot)$ označuje číslo podmíněnosti.

Z nerovnosti (2.30) plyne, že pokud jsou vektory Lanczosovy báze dobře podmíněny, jsou normy reziduí metody QMR blízké normám optimálních reziduí metody GMRES.

Věta 2.5 nám tak do značné míry dává návod, jak se lze postavit k otázce optimálního stínového vektoru. Za optimální stínový vektor můžeme například považovat takový startovací vektor $\widetilde{r}_0$, pro který je Lanczosova báze nejlépe podmíněná. Podrobněji, uvažujme

libovolnou třídiagonální matici $\mathbf{T}$ s nenulovými poddiagonálními i naddiagonálními prvky, která je podobná matici $\mathbf{A}$ a sloupce matice transformace $\mathbf{V}$ nechť mají normu 1, platí

$$\mathbf{A} = \mathbf{V}\mathbf{T}\mathbf{V}^{-1}. \tag{2.31}$$

Provedeme-li $\mathbf{QR}$ rozklad matice $\mathbf{V}$,

$$\mathbf{V} = \mathbf{Q}\mathbf{R}, \tag{2.32}$$

tvoří prvních k sloupců matice $\mathbf{Q}$ ortonormální bázi Krylovova podprostoru $\mathcal{K}_k(\mathbf{A}, r_0)$ $(1 \leq k \leq n)$ a z jednoznačnosti určení je lze považovat za vektory určené Arnoldiho procesem, platí

$$\mathbf{A} = \mathbf{Q}\mathbf{H}\mathbf{Q}^T, \tag{2.33}$$

kde $\mathbf{H}$ je horní Hessenbergova matice z Arnoldiho procesu. Z maticových rovnic (2.31), (2.32) a (2.33) plyne vztah

$$\mathbf{H} = \mathbf{R}\mathbf{T}\mathbf{R}^{-1}. \tag{2.34}$$

Původní problém nalezení nejlépe podmíněné transformace matice $\mathbf{A}$ na třídiagonální matici jsme uvedeným postupem převedli na problém nalezení nejlépe podmíněné horní trojúhelníkové matice $\mathbf{R}$, která transformuje horní Hessenbergovu matici na třídiagonální tvar. Poznamenejme, že vzhledem k (2.32) platí

$$\kappa(\mathbf{V}) = \kappa(\mathbf{Q}\mathbf{R}) = \kappa(\mathbf{R})$$

a matice $\mathbf{R}$ má navíc všechny sloupce normovány na jedničku. Pokud tedy budeme schopni nalézt optimální transformaci pro problém (2.34), budeme téhož schopni i pro problém (2.31). Na problém (2.34) samozřejmě můžeme opět nahlížet jako na Lanczosův proces se startovacím parametrem $v_1 = e_1$ a naším cílem je určit stínový vektor (1. řádek matice $\mathbf{R}^{-1}$) takový, aby byla Lanczosova báze (sloupce matice $\mathbf{R}$) co nejlépe podmíněna. S řešením tohoto problému lze například začít tak, že uvažujeme matici $\mathbf{H}$ v nějakém jednodušším tvaru, např. třídiagonální s navíc jedním nenulovým prvkem v horním trojúhelníku, a snažíme se najít optimální stínový vektor (který vede na nejlépe podmíněnou Lanczosovu bázi) pro tento případ.

Jednou z další možností definice optimálního stínového vektoru je volba vedoucí k nejbližším reziduovým konvergenčním křivkám QMR a GMRES, přičemž blízkost může být měřena různými způsoby (např. odchylkami (úhly) mezi uvažovanými vektory resp. kosiny těchto úhlů, či logaritmickou vzdáleností norem příslušných reziduí). V každém případě se zdá, že hledání optimálního stínového vektoru nebude jednoduchou záležitostí a pravděpodobně povede na úlohu optimalizace (hledání minima nelineárního funkcionálu). Otevřeným problémem je rovněž geometrická interpretace optimálního stínového vektoru, která je důležitá především pro pochopení fungování nesymetrického Lanczosova procesu ale i pro případné speciální praktické aplikace, v nichž budeme znát potřebné údaje.

2.5 Metody s krátkými rekurencemi a pravděpodobnost?

Při numerických experimentech občas pozorujeme, že se konvergenční křivky metod s krátkými rekurencemi (LM a QMR) velmi málo liší od konvergenční křivky optimální GMRES. Počítáme-li experimenty v násobné aritmetice (například proto, abychom se přesvědčili o vlivu zaokrouhlovacích chyb), zjistíme, že je tento jev dokonce velmi častý. Zvláště

v situaci, kdy konvergenční křivka GMRES začne rychle klesat, jsou konvergenční křivky ostatních klasických krylovovských metod (LM, QMR, FOM) velmi blízké konvergenční křivce GMRES. V této sekci se pokusíme vysvětlit jeden z důvodů, proč tento jev nastává.

Krylovovské metody postupně vytvářejí posloupnost prostorů $\mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$ a hledají k-té reziduum ve varietě $r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$. Metoda GMRES přitom určuje reziduum r_k^G s nejmenší možnou normou z této variety, $r_k^G \perp \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$. Zároveň platí

$$r_k^G \cup \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0) = \mathcal{K}_{k+1}(\mathbf{A}, r_0),$$

r_k^G je ortogonálním doplňkem prostoru $\mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$ v prostoru $\mathcal{K}_{k+1}(\mathbf{A}, r_0)$ a jeho norma udává vzdálenost variety $r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$ od počátku. Otázka rychlosti konvergence metody GMRES je otázkou rychlosti přibližování variety $r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$ k počátku. Jinak řečeno, varieta $r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$ postupně pohlcuje počáteční reziduum r_0 (zmenšuje se úhel mezi reziduem r_0 a prostorem $\mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$) a rychlost konvergence je rychlostí pohlcování rezidua r_0.

Klasické krylovovské metody (LM, QMR, FOM, GMRES) mají společnou tu vlastnost, že jejich k-té reziduum doplňuje varietu $r_0+\mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$ na prostor $\mathcal{K}_{k+1}(\mathbf{A}, r_0)$. Označme $k+1$ rozměrnou jednotkovou kouli v prostoru $\mathcal{K}_{k+1}(\mathbf{A}, r_0)$ symbolem Ω_{k+1} (každý vektor z $\mathcal{K}_{k+1}(\mathbf{A}, r_0)$ potom dostaneme jako násobek jednotkového vektoru z této koule). Uvažujme nyní teoretickou krylovovskou metodu, která vezme náhodný vektor (předpokládáme rovnoměrné rozdělení) z koule Ω_{k+1} a reziduum r_k definuje jako průnik přímky určené tímto vektorem s varietou $r_0+\mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$. Nechť $\gamma > 1$ je nějaké číslo. Zajímá nás, s jakou pravděpodobností pro takto náhodně zvolený reziduový vektor r_k platí nerovnost

$$\|r_k\| \leq \gamma \|r_k^G\|. \tag{2.35}$$

Koncové body všech vektorů reziduí r_k, jež vyhovují podmínce (2.35) zřejmě leží v k-rozměrné kouli $\Gamma_k \in r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$ se středem v bodě r_k^G (koncový bod vektoru r_k^G) a poloměrem

$$\rho_k = \sqrt{\gamma^2 \|r_k^G\|^2 - \|r_k^G\|^2} = \|r_k^G\| \sqrt{\gamma^2 - 1}.$$

Označme α maximální odchylku rezidua r_k od rezidua r_k^G, pro kterou nastává v (2.35) rovnost. Celou situaci znázorňujeme na obrázku 2.1. Z obrázku 2.1 je zřejmé, že pravděpodobnost, se kterou se odchýlí přímka určená náhodně zvoleným vektorem z jednotkové koule Ω_{k+1} od směru vektoru r_k^G nejvýše o úhel α, lze vyjádřit následovně

$$P(\|r_k\| \leq \gamma \|r_k^G\|) = \frac{4\alpha}{2\pi} = \frac{2\alpha}{\pi}, \tag{2.36}$$

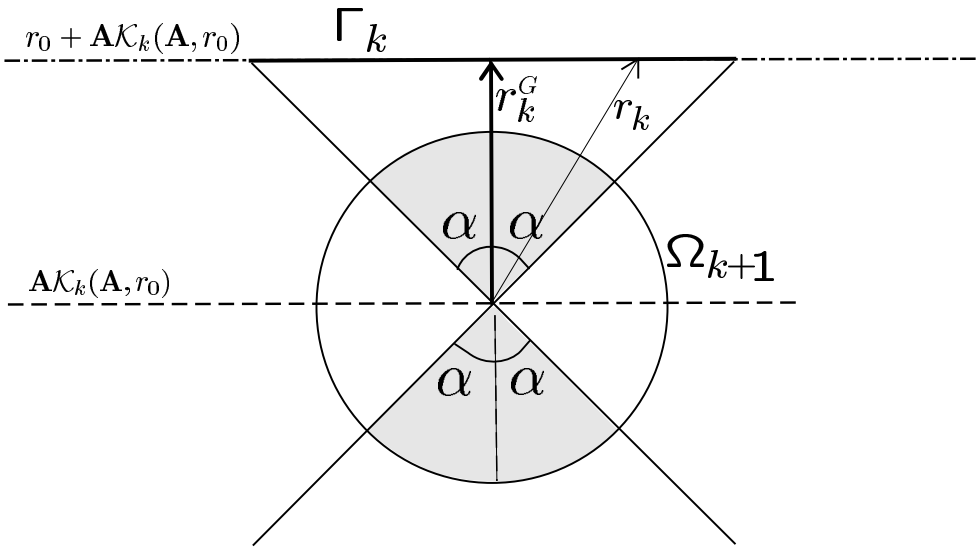
kde $P(Q)$ označuje pravděpodobnost s jakou nastane náhodný jev Q. Kosinus úhlu mezi vektory r_k^G a r_k roven podílu $\|r_k^G\|/\|r_k\|$ a pro maximální odchylku α dostáváme

$$\cos(\alpha) = \frac{\|r_k^G\|}{\gamma \|r_k^G\|} = \gamma^{-1}. \tag{2.37}$$

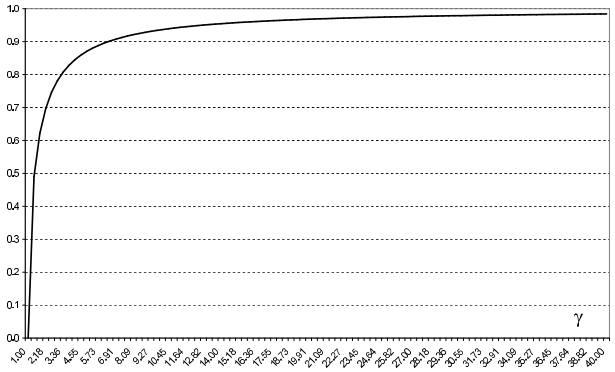
Pravděpodobnost (2.36) lze s využitím (2.37) vyjádřit ve tvaru

$$P(\|r_k\| \leq \gamma \|r_k^G\|) = \frac{2 \arccos(\gamma^{-1})}{\pi}. \tag{2.38}$$

Kupříkladu, pro $\gamma = 10$ je tato pravděpodobnost rovna $93,6\%$, pro hodnotu $\gamma = 100$ $99,4\%$ atd., viz. obrázek 2.2. Zformulovali jsme jeden z důvodů, proč je konvergenční křivka např. LM často velmi blízká konvergenční křivce GMRES. U metody LM samozřejmě není k-té reziduum voleno jako náhodný vektor z jednotkové koule, ale je kolmé



Obrázek 2.1: Náhodně zvolené reziduum r_k a reziduum r_k^G



Obrázek 2.2: Pravděpodobnost, že $\|r_k\| \leq \gamma \|r_k^G\|$

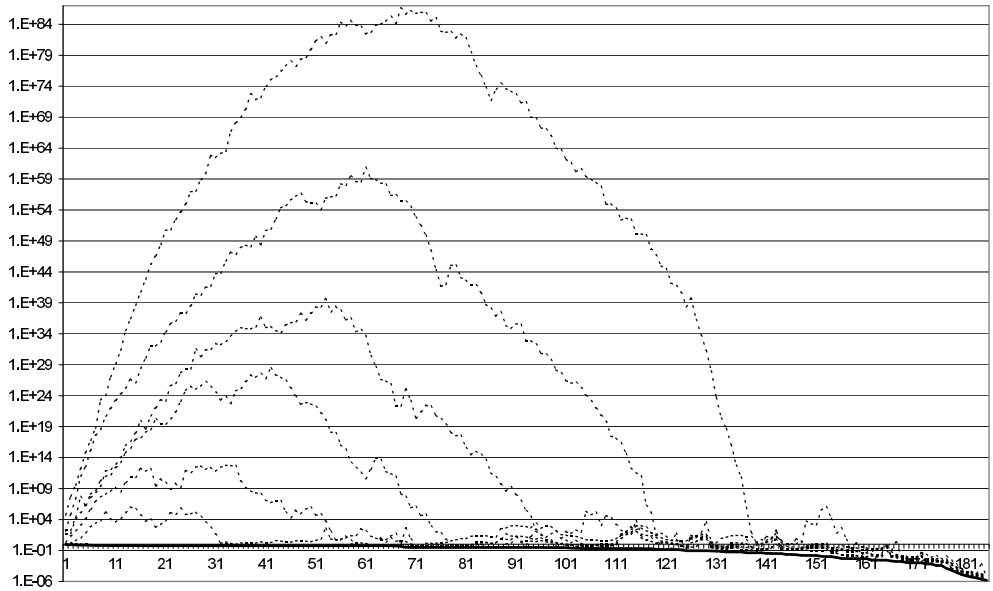
na Krylovův podprostor $\mathcal{K}_k(\mathbf{A}^T, \widetilde{r}_0)$ a úhel mezi reziduem r_k^L a r_k^G zřejmě závisí na odchylce prostorů $\mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$ a $\mathcal{K}_k(\mathbf{A}^T, \widetilde{r}_0)$. Odchylka těchto dvou prostorů není náhodná, je dána vlastnostmi matice $\mathbf{A}$ a vektory r_0 a $\widetilde{r}_0$, nicméně, chování LM je často srovnatelné s teoretickou krylovovskou metodou, která za reziduum volí vhodně normovaný náhodný vektor z jednotkové koule. Důležitou roli zde samozřejmě hraje i rychlost přibližování variety $r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$ k počátku. Pokud je tato rychlost nízká, začíná se kromě pravděpodobnostního principu (2.38) projevovat i způsob, jakým jsou určeny reziduové vektory dané metody a v těchto iteracích se mohou rezidua ostatních krylovovských metod velmi odchýlit od reziduí metody GMRES. Konkrétněji, pokud je varieta $r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$ blízká varietě $r_0 + \mathbf{A}\mathcal{K}_{k+1}(\mathbf{A}, r_0)$, je i reziduum r_k^G blízké reziduu r_{k+1}^G. Je-li odchylka reziduí r_k^F (reziduum metody FOM) a r_k^G malá (t.j. i odchylka mezi r_k^F a r_{k+1}^G je malá), je nutně odchylka mezi r_{k+1}^G a r_{k+1}^F velká, neboť r_{k+1}^F je kolmé na r_k^F (a to je blízké r_{k+1}^G).

Pravděpodobnostní princip (2.38) nám rovněž umožňuje komentovat věty 2.1–2.4. Zde jsme narazili na problém „náhodná tříkroková rekurence versus náhodný stínový vektor“. Měli bychom si uvědomit, že pokud volíme koeficienty tříkrokové rekurence náhodně (to můžeme dělat až do kroku $n/2$ a vytvářená rezidua stále můžeme považovat za rezidua

metody LM), konstruujeme v podstatě náhodná rezidua z celé variety $r_0 + \mathbf{A}\mathcal{K}_k(\mathbf{A}, r_0)$ a není žádného důvodu, proč by normy takto vytvářených reziduí měly klesat. Pokud však volíme náhodně stínový vektor, vnášíme jistý prvek náhody do vytvářeného prostoru $\mathcal{K}_k(\mathbf{A}^T, \widetilde{r}_0)$ a tím i do odchylky reziduí r_k^L a r_k^G. Lze očekávat, že se v Lanczosově metodě s náhodně voleným stínovým vektorem budou projevovat vlastnosti teoretické krylovovské metody, která za reziduum volí vhodně normovaný náhodný vektor z jednotkové koule.

Podle našeho názoru jsou, z teoretického hlediska, metody s krátkými rekurencemi velmi efektivní, protože s malými výpočetními náklady velmi často určují dobré aproximace řešení (blízké aproximacím metody GMRES), kdykoliv konvergenční křivka metody GMRES dostatečně rychle klesá. Z praktického hlediska jsou fundamentálním problémem metod s krátkými rekurencemi zaokrouhlovací chyby, které mohou způsobit zpoždění konvergence ale i úplné znehodnocení výpočtu. Problém spočívá v tom, že nepřesně vypočtený výsledek z jedné iterace dosazujeme jako vstupní data do další iterace. Tím se samozřejmě celková chyba násobí a dochází k jejímu exponenciálnímu nárůstu a k porušení základních vlastností (např. ortogonality), na kterých jsou metody postaveny. Pokud chceme metody s krátkými rekurencemi používat a spoléhat se na ně, bude nutné pochopit jejich chování v konečné aritmetice počítače a určit případy, kdy je lepší (z důvodů zaokrouhlovacích chyb) použít pomalejší, ale robustnější, metody s dlouhými rekurencemi. Motivací a návodem nám do jisté míry může být chování metody sdružených gradientů (CG) v konečné aritmetice (matice $\mathbf{A}$ je symetrická a pozitivně definitní), kterému se na základě prací [22], [28], [57], [59] věnujeme v kapitole 4.

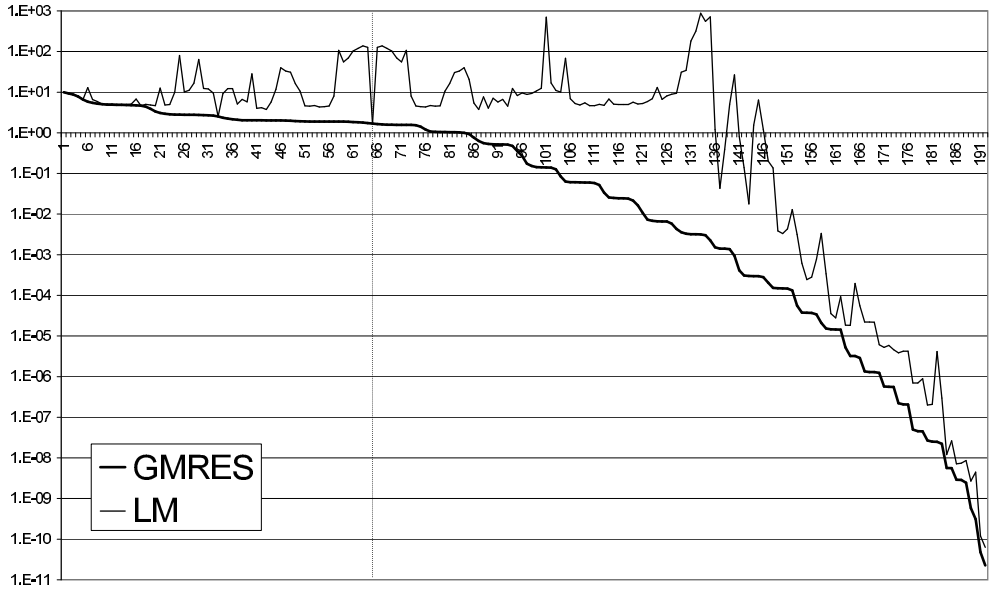
2.6 Numerické experimenty



Obrázek 2.3: Výpočet reziduí GMRES pomocí LM (matice STEAM1)

Pro první experiment jsme uvažovali matici STEAM1 (Harwell-Boeing Collection, Matrix Market [37]) řádu 240, náhodný vektor pravé strany b a počáteční přiblížení x_0 rovné nulovému vektoru. Stínový vektor $\widetilde{r}_0$ jsme volili jako ortogonální projekci vektoru pravé strany $b\,(= r_0)$ na prostor $\mathcal{K}_k(\mathbf{A}^T, r_k^G)$, kde r_k^G je reziduum metody GMRES. Pokud platí $r_0 \notin \mathcal{K}_k(\mathbf{A}^T, r_k^G)$, splňuje zřejmě takto zvolený vektor $\widetilde{r}_0$ podmínku (2.20) a nedojde-li k

předčasnému ukončení algoritmu, který realizuje Lanczosovu metodu (v našem případě algoritmu BiCG), je k-té reziduum LM shodné s k-tým reziduem GMRES. Na obrázku 2.3 vykreslujeme plnou křivkou konvergenční reziduovou křivku metody GMRES. Ostatní křivky reprezentují konvergenční reziduové křivky Lanczosovy metody s různými stínovými vektory, které splňují podmínku (2.20) postupně pro $k = 20, 40, 60, \ldots, 140$. Obrázek 2.3 demonstruje, že sice je možné určit stínový vektor vedoucí k rovnosti rezidua LM a GMRES, avšak tento výpočet může být numericky velmi nestabilní (jak je vidět ze stále se zvyšujících převýšení v konvergenčních křivkách LM). Abychom byli schopni vypočíst zvolená rezidua GMRES pomocí LM, byli jsme nuceni použít násobné aritmetiky (balík MPFUN [5]) a výpočty vyčíslovat na 500 platných cifer. Nicméně, uvedený numerický experiment potvrdil naše teoretické výsledky.



Obrázek 2.4: Výpočet více reziduí GMRES pomocí LM (matice `CAVITY01`)

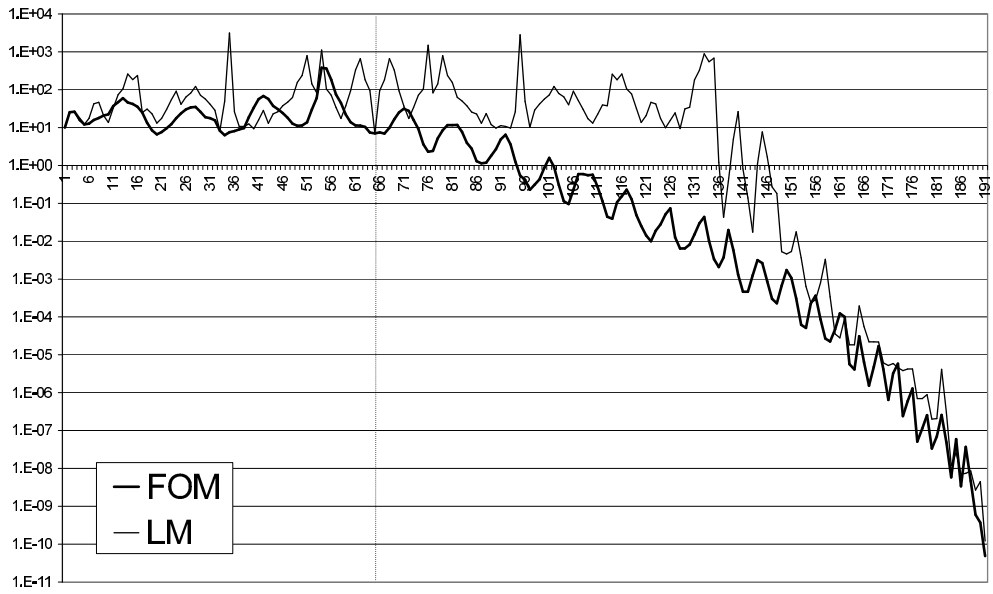
Pro další experiment jsme uvažovali matici `CAVITY01` (SPARSKIT Collection, Matrix Market [37]) řádu 317, náhodný vektor pravé strany b a počáteční přiblížení x_0 rovné nulovému vektoru. Abychom odhlédli od vlivu konečné aritmetiky počítače (jde nám o teoretické výsledky), použili jsme opět násobné aritmetiky (balík MPFUN [5]) a výpočty vyčíslovali na 500 platných cifer. Na obrázcích 2.4 a 2.5 demonstrujeme, že lze určit stínový vektor takový, aby LM vypočetla vybraná rezidua jiné krylovovské metody.

Na obrázku 2.4 jsme uvažovali metodu GMRES (tučná plná křivka) a stínový vektor jsme volili jako ortogonální projekci vektoru pravé strany na prostor $\mathcal{Z}_{64}$ (viz. definice (2.24)) generovaný maticí $\mathbf{A}$ a rezidui metody GMRES (reziduová křivka LM je vykreslena plnou křivkou normální tloušťky). Je vidět, že při této volbě stínového vektoru skutečně platí

$$r_0^L = r_0^G,\ r_1^L = r_1^G,\ r_2^L = r_2^G,\ r_4^L = r_4^G,\ \ldots,\ r_{64}^L = r_{64}^G.$$

Navíc můžeme pozorovat zajímavý jev. Konvergenční křivka LM v iteracích $1, \ldots, 64$ je téměř symetrická podle svislé osy vedoucí 64-tou iterací (tenká čára) s konvergenční křivkou LM v iteracích $64, \ldots, 128$.

Obrázek 2.5 je analogií obrázku 2.4 pro metodu FOM (tučná plná křivka). Rezidua Lanczosovy metody se startovacím parametrem $\widetilde{r}_0 \perp \mathcal{Z}_{64}$, kde $\mathcal{Z}_{64}$ je generován maticí $\mathbf{A}$ a rezidui metody FOM, jsou pro $k = 0, 1, 2, 4, \ldots, 64$ shodná s rezidui metody FOM.



Obrázek 2.5: Výpočet více reziduí FOM pomocí LM (matice `CAVITY01`)

Opět můžeme pozorovat symetrii, která nastane v konvergenční křivce LM podle svislé osy procházející 64-tou iterací.

Než uvedeme další numerický experiment, věnujme se krátce otázce náhodného stínového vektoru. V praxi bývá častý přístup k získání náhodného vektoru takový, že se použije procedura daného programovacího jazyka, která poskytuje „náhodná“ čísla (generátor náhodných čísel) a jednotlivé složky požadovaného vektoru se nagenerují pomocí této procedury. První otázka, kterou bychom si ale měli položit je, s jakým pravděpodobnostním rozdělením poskytuje generátor náhodná čísla. Zřejmě bychom měli použít takový generátor, který generuje čísla z intervalu $(0, 1)$ se stejnou pravděpodobností, t.j s rovnoměrným rozdělením. Pokud však dosazujeme do jednotlivých složek vektoru takto získaná čísla (z generátoru náhodných čísel s rovnoměrným rozdělením), obdržíme náhodný vektor z jednotkové krychle a zřejmě získáme s větší pravděpodobností vektor, který je blízko rohu této krychle (je tam více hmoty) a s menší pravděpodobností vektor, který je blízko středu stěny krychle. Ideální by zřejmě bylo, kdybychom dokázali generovat náhodné vektory s rovnoměrným rozdělením z jednotkové koule (resp. jednotkové sféry). To lze ovšem provést velmi jednoduše následujícím způsobem.

Každý vektor $w \in \mathbb{R}^n$, $w = (w_1, \ldots, w_n)^T$ jednotkové sféry lze vyjádřit pomocí sférických souřadnic

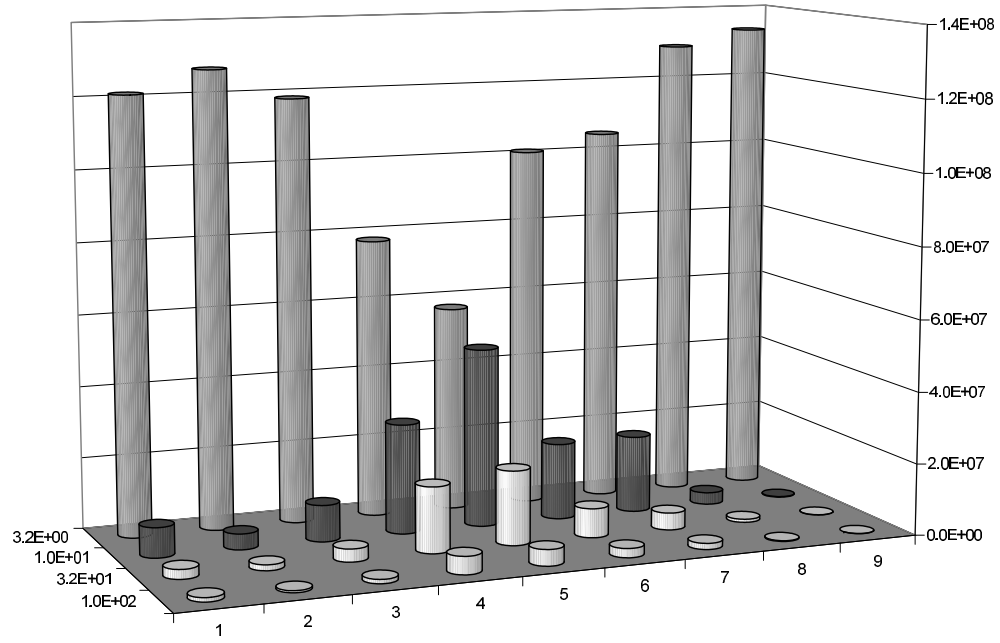
$$\begin{array}{lllllll}
w_1 & = & \cos(\vartheta_1) & \cos(\vartheta_2) & \cos(\vartheta_3) & \ldots & \cos(\vartheta_{n-1}), \\
w_2 & = & & \cos(\vartheta_2) & \cos(\vartheta_3) & \ldots & \cos(\vartheta_{n-1}) \quad \sin(\vartheta_1), \\
w_3 & = & & & \cos(\vartheta_3) & \ldots & \cos(\vartheta_{n-1}) \quad \sin(\vartheta_2), \\
\vdots & = & & & & \ddots & \vdots \qquad\quad \vdots \\
w_{n-1} & = & & & & & \cos(\vartheta_{n-1}) \quad \sin(\vartheta_{n-2}), \\
w_n & = & & & & & \sin(\vartheta_{n-1}),
\end{array}$$

kde $\vartheta_1 \in \langle -\pi/2, \pi/2), \ldots, \vartheta_{n-2} \in \langle -\pi/2, \pi/2)$, $\vartheta_{n-1} \in \langle 0, 2\pi)$. Protože nezáleží na znaménku stínového vektoru, stačí vzít pouze polovinu jednotkové sféry, t.j budeme uvažovat všechny středové úhly ϑ_i z intervalu $\langle -\pi/2, \pi/2)$. Nyní stačí použít generátor náhodných

čísel s rovnoměrným rozdělením na intervalu $\langle -\pi/2, \pi/2)$, vygenerovat náhodné úhly ϑ_1, $\ldots$, ϑ_{n-1} a vypočíst hodnoty $w_1, \ldots, w_n$. Tím získáme náhodný vektor z jednotkové sféry.

Výpočet souřadnic náhodného vektoru lze provést efektivně například pomocí následující procedury zapsané v jazyce FORTRAN. Vstupními parametry této procedury jsou náhodný vektor úhlů `TH` $[= (\vartheta_1, \ldots, \vartheta_{n-1})^T]$, dimenze `N` $[= n]$ a výstupním parametrem je náhodný vektor `W` $[= w]$.

```
      SUBROUTINE UNIT_SPHERE(TH,N,W)
C
      integer N
      double precision TH(*), W(*)
C
      integer i
C
      W(N) = 1.0
      do i = N-1, 1, -1
        W(i) = COS(TH(i))
        W(i) = W(i)*W(i+1)
      end do
      do i = 2, N
        W(i)=W(i)*SIN(TH(i-1))
      end do
      END
```



Obrázek 2.6: Rozložení konvergenčních křivek LM

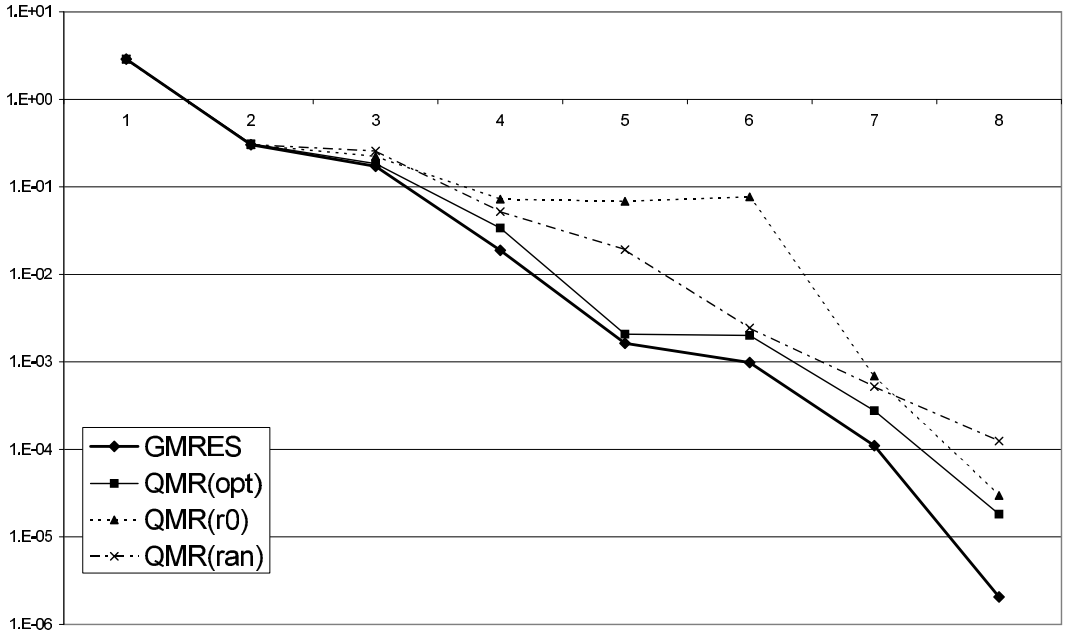
K následujícímu experimentu použijme matici $\mathbf{A} \in \mathbb{R}^{n\times n}$, $n = 10$ vytvořenou příkazem

$$\mathbf{A} = \mathrm{eye}(n, n) + 0.1 * \mathrm{triu}(\mathrm{randn}(n, n)) \tag{2.39}$$

v systému MATLAB, viz. rovněž [26] (příkaz eye(n,n) definuje matici identity a příkaz triu(randn(n,n)) vytvoří horní trojúhelníkovou matici z náhodné matice). Za vektor pravé strany volme náhodný vektor a za počáteční přiblížení nulový vektor. Matice $\mathbf{A}$ počítaná podle předpisu (2.39) se vyznačuje špatně podmíněnou bází vlastních vektorů, konkrétněji $\kappa(\mathbf{Z}) = 6.2 \times 10^3$, kde $\mathbf{Z}$ je matice s normovanými vlastními vektory matice $\mathbf{A}$ ve svých sloupcích. Dodejme, že $\kappa(\mathbf{A}) = 2.0 \times 10^2$. Rozdělme interval $\langle -\pi/2, \pi/2)$ na $m = 8$ stejných intervalů a jejich pravé krajní body označme $\pi_1, \ldots, \pi_m$. Množinu všech vektorů z n-rozměrné jednotkové sféry, pro které úhly ϑ_i nabývají pouze hodnot $\pi_1, \ldots, \pi_m$ označme $\mathbf{P}_m^{(n)}$. Za stínové vektory budeme volit všechny vektory z této množiny rovnoměrně rozložených vektorů jednotkové sféry. Poznamenejme, že jejich počet je roven

$$m^{n-1} = 8^9 \sim 1.3 \times 10^8.$$

Uvažujme všechny reziduové konvergenční křivky LM nastartované se stínovými vektory z množiny $\mathbf{P}_8^{(10)}$. Zajímá nás odlišnost většiny konvergenčních křivek LM od konvergenční křivky GMRES. Výsledky našeho experimentu zobrazujeme na obrázku 2.6. Osy na obrázku (2.6) mají následující význam: na osu x nanášíme číslo iterace $(1, \ldots, 9)$, na osu y vzdálenost od reziduové konvergenční křivky GMRES a na osu z počet reziduí r_k^L metody LM, jejichž norma v k-té iteraci (osa x) leží v určité vzdálenosti (osa y) od normy rezidua GMRES. Vidíme, že většina norem reziduí LM leží v blízkosti norem reziduí GMRES, platí pro ně $\|r_k^L\| \leq 3.2\,\|r_k^G\|$. Ve středu iterační křivky je nezanedbatelný počet reziduí LM více vzdálen od konvergenční křivky GMRES (pravděpodobně je to důsledek špatně podmíněné báze vlastních vektorů; soudíme tak z obdobného experimentu s maticí s dobře podmíněnou bází vlastních vektorů, ve kterém tento jev nenastal a drtivá většina norem reziduí LM byla v těsné blízkosti norem reziduí GMRES). Směrem k n-té iteraci se opět téměř všechny konvergenční křivky LM těsně přimknou ke konvergenční křivce GMRES.



Obrázek 2.7: Numericky vypočtený optimální stínový vektor

V posledním numerickém experimentu jsme se pokusili numericky vypočíst optimální stínový vektor $\widetilde{r}_0^O$ (resp. jeho aproximaci), přičemž za optimální stínový vektor jsme po-

važovali argument minima funkcionálu

$$f(\mathbf{A}, b, x_0, \widetilde{r}_0) \equiv \sum_{i=1}^{n-1} \Big(\log(\|r_i^Q\|) - \log(\|r_i^G\|) \Big)^2 = \sum_{i=1}^{n-1} \log^2 \left(\frac{\|r_i^Q\|}{\|r_i^G\|} \right) \tag{2.40}$$

pro pevné $\mathbf{A}$, b a x_0, kde r_i^Q označuje reziduum metody QMR a r_i^G reziduum metody GMRES. Použili jsme opět matici (2.39), avšak pro $n = 8$. Zvolili jsme $m = 16$ (počet dělení každého úhlu) a za $\widetilde{r}_0$ jsme postupně dosazovali všechny vektory z množiny $\mathbf{P}_{16}^{(8)}$. Z vypočtených hodnot funkcionálu $f(\mathbf{A}, b, x_0, \widetilde{r}_0)$ jsme určili tu minimální.

Na obrázku 2.7 vidíme, že reziduová křivka optimální QMR (plná křivka normální tloušťky) může být velmi blízká konvergenční křivce GMRES (plná tučná křivka) a v podstatě kopíruje tvar křivky GMRES. Dále jsme pro porovnání znázornili konvergenční křivku QMR, přičemž za stínový vektor jsme volili počáteční reziduum (čárkovaná křivka) a náhodně volený stínový vektor z jednotkové koule (čerchovaná křivka).

3

Odhady **A**-normy chyby v CG

Metoda sdružených gradientů (CG) *slaví své 50-té narozeniny. Byla srozumitelně popsána již svými tvůrci Hestenesem a Stiefelem* [*32*]*, kteří odvodili všemožné algebraické vztahy mezi vektory a koeficienty algoritmu, ukázali souvislost* CG *s Gaussovou kvadraturou a minimalizací funkcionálu. Hestenes a Stiefel poukázali na to, že norma rezidua není spolehlivá konvergenční charakteristika a za vhodného kandidáta na určení kvality počítané aproximace označili* **A***-normu chyby. Důležitost* **A***-normy chyby byla potvrzena i aplikacemi ve fyzice a kvantové chemii* (*zde bývá* **A***-norma chyby nazývána energetickou normou*)*. Hodnotu* **A***-normy chyby však nelze obecně přesně určit, pouze odhadnout. K odhadům bylo použito vztahu mezi* CG *a Gaussovou kvadraturou a v poslední době začaly vznikat odhady založené na jednoduché algebraické manipulaci. Budeme prezentovat nový odhad* **A***-normy chyby odvozený algebraickou cestou a ukážeme, že nejjednodušší a numericky nejstabilnější odhad je skryt již v původní práci* [*32*] (*numerickou stabilitou odhadu se budeme zabývat v kapitole 5*)*. Vysvětlíme, že odhady* **A***-normy v publikacích* [*8*]*,* [*17*]*,* [*19*]*,* [*18*]*,* [*38*] *odvozené pomocí Gaussovy kvadratury jsou matematicky ekvivalentní odhadům odvozeným algebraickou cestou.*

Naše vyprávění bychom měli začít v roce 1952. Tehdy pánové Hestenes a Stiefel [32] sepsali článek, který se zapsal do dějin, zformulovali metodu sdružených gradientů (CG), velmi silný a nenáročný nástroj pro řešení soustav lineárních rovnic se symetrickou, pozitivně definitní maticí. Vyčerpávajícím způsobem popsali vztahy mezi vektory algoritmu, souvislost s minimalizací funkcionálu a Gaussovou kvadraturou. Byli si vědomi toho, že s konečnou aritmetikou počítačů přicházejí další netriviální problémy. Při podrobném čtení tohoto článku zjistíme, že některé jejich myšlenky, publikované v roce 1952, doznávají pochopení až dnes, po padesáti letech.

Při běhu metody sdružených gradientů potřebujeme nějakou informaci o kvalitě počítané aproximace. Jedním z kandidátů na získání této informace je reziduum, počítané v každém kroku metody. V [32, str. 410] se píše, že obvykle může být norma tohoto vektoru použita jako míra „kvality“ pro korespondující aproximaci a tato věta se zřejmě nejvíce zapsala do povědomí dalších autorů. Hodnocení rezidua však v [32, str. 410] dále pokračuje: Nicméně, tato míra není spolehlivá, jelikož je možné zkonstruovat příklady, kdy norma rezidua roste v každém kroku. Místo toho zavádí v [32, str. 413] chybovou funkci (kvadrát **A**-normy chyby), kterou považují za vhodného kandidáta na měření kvality aproximace. Hodnoty této funkce nelze obecně přesně určit, pouze odhadnout. Hestenes a Stiefel neuvedli žádnou proceduru na zastavování algoritmu. Avšak vztah mezi kvadráty **A**-norem chyb, který prezentují [32, (6:2)], může být přímo použit, jak ukážeme, k odhadu **A**-normy chyby a tím i ke konstrukci zastavovací procedury.

Myšlenka odhadování normy chyby v CG, která hraje významnou roli při hodnocení konvergence viz. [1], [2], byla vzkříšena a rozšířena pracemi Goluba a jeho spolupracovníků. V [8] autoři vztahují (s použitím starších výsledků, viz. reference v [8]) normu chyby k Gaussově kvadratuře (a k jejím modifikacím). Pojetí uvedené v této práci se

stalo základem pozdějšího vývoje. Odhad normy chyby v iteračních metodách byl intenzivně studován v mnoha pozdějších pracech, viz. např. [11], [17], [19], [18], [38], [39], [7]. V [4] autoři uvažovali předpodmíněnou metodu sdružených gradientů a uvedli (znovuobjevili) [32, (6:2)] nezávisle na [32]. Bylo experimentálně pozorováno, že numerické hodnoty dané výsledným odhadem jsou identické s hodnotami obdrženými z odhadu založeného na Gaussově kvadratuře, shoda však nebyla v [4] nijak vysvětlena.

Soustředíme se na dolní odhady **A**-normy chyby. Obsah této kapitoly je součástí našeho článku [59] a je založen dřívějších pracech [41], [57] a dalších viz. citace v [59].

Kapitola je strukturována následujícím způsobem. Sekce 3.1 uvádí čtenáře do problému, definujeme algoritmus CG a symetrický Lanczosův algoritmus, které mohou být lehce odvozeny na základě výsledků z první kapitoly, a ukazujeme jejich vztah k Gaussově kvadratuře. V 3.2 užijeme k vyjádření výsledků Gaussovy kvadratury hodnoty, jež máme k dispozici z algoritmu CG. Tím získáme několik podob Gaussova kvadraturního vzorce, které jsou všechny matematicky ekvivalentní a jež užijeme v 3.3 ke konstrukci odhadu **A**-normy chyby v metodě CG.

3.1 Metoda sdružených gradientů a Gaussova kvadratura

Uvažujme systém lineárních rovnic

$$\mathbf{A}x = b,$$

kde $\mathbf{A} \in \mathbb{R}^{n\times n}$ je reálná symetrická pozitivně definitní matice a $b \in \mathbb{R}^n$ vektor pravé strany. Nechť x_0 je počáteční aproximace řešení. Základní myšlenka metody sdružených gradientů (CG) spočívá v tom, že určuje j-tou aproximaci řešení podmínkou

$$\begin{aligned} x_j &\in x_0 + \mathcal{K}_j(\mathbf{A}, r_0) \\ \|x - x_j\|_{\mathbf{A}} &= \min_{u\in x_0+\mathcal{K}_j(\mathbf{A},r_0)} \|x - u\|_{\mathbf{A}}, \end{aligned} \tag{3.1}$$

t.j. minimalizuje **A**-normu chyby $\|x-x_j\|_{\mathbf{A}} \equiv \big((x-x_j), \mathbf{A}(x-x_j)\big)^{1/2}$ přes všechny vektory z variety $x_0 + \mathcal{K}_j(\mathbf{A}, r_0)$, kde

$$\mathcal{K}_j(\mathbf{A}, r_0) = \mathrm{span}\{r_0, \mathbf{A}r_0, \dots \mathbf{A}^{j-1}r_0\}$$

je j-tý Krylovův prostor generovaný maticí **A** a počátečním reziduem r_0, $r_0 = b - \mathbf{A}x_0$. Standardní implementace CG, která byla uvedena v [32, (3:1a)-(3:1f)] a kterou prezentujeme jako algoritmus 3.1, může být lehce odvozena z algoritmu 1.6 BiCG, uvážíme-li zjednodušení v rekurencích, která plynou ze symetrie matice **A** a z volby $\widetilde{r}_0 = r_0$. Připomeňme, že vektory reziduí $\{r_0, r_1, \dots, r_{j-1}\}$ tvoří ortogonální bázi a směrové vektory $\{p_0, p_1, \dots, p_{j-1}\}$ **A**-ortogonální bázi j-tého Krylovova prostoru $\mathcal{K}_j(\mathbf{A}, r_0)$.

Důležitou úlohu v našem výkladu bude hrát blízký vztah mezi CG a symetrickým Lanczosovým algoritmem [36]. Symetrický Lanczosův algoritmus 3.2 (SL) (jeho podobu lze odvodit z algoritmu 1.2 NL) počítá pro matici **A** a vektor r_0 posloupnost ortonormálních vektorů $v_1, v_2, \dots$. Nechť $\mathbf{V}_j = [v_1, \dots, v_j]$ značí matici o rozměrech $n \times j$, jež má ve svých sloupcích Lanczosovy vektory $\{v_1, \dots, v_j\}$ a nechť $\mathbf{T}_j$ je symetrická třídiagonální matice ve tvaru

$$\mathbf{T}_j = \begin{pmatrix} \alpha_1 & \beta_2 & & \\ \beta_2 & \alpha_2 & \ddots & \\ & \ddots & \ddots & \beta_j \\ & & \beta_j & \alpha_j \end{pmatrix}. \tag{3.2}$$

Algoritmus 3.1. *Metoda sdružených gradientů* (CG)

input $x_0,\ \mathbf{A},\ b$

initialization

$$\begin{aligned} r_0 &= b - \mathbf{A}x_0 \\ p_0 &= r_0 \end{aligned}$$

for $j = 0, 1, \ldots$

$$\gamma_j = \frac{(r_j, r_j)}{(p_j, \mathbf{A}p_j)}$$

$$\begin{aligned} x_{j+1} &= x_j + \gamma_j\, p_j \\ r_{j+1} &= r_j - \gamma_j\, \mathbf{A}p_j \end{aligned}$$

$$\delta_j = \frac{(r_{j+1}, r_{j+1})}{(r_j, r_j)}$$

$$p_{j+1} = r_{j+1} + \delta_j\, p_j$$

end for

Algoritmus 3.2. *Symetrický Lanczosův algoritmus* (SL)

input $\mathbf{A},\ r_0$

initialization

$$\begin{aligned} v_0 &= \mathbf{o} \\ v_1 &= r_0/\|r_0\| \\ \beta_1 &= 0 \end{aligned}$$

for $j = 1, 2, \ldots$

$$\alpha_j = (\mathbf{A}v_j - \beta_j v_{j-1}, v_j)$$

$$\begin{aligned} w_j &= \mathbf{A}v_j - \alpha_j v_j - \beta_j v_{j-1} \\ \beta_{j+1} &= \|w_j\| \\ v_{j+1} &= w_j/\beta_{j+1} \end{aligned}$$

end for

Potom lze psát algoritmus 3.2 v maticové podobě

$$\mathbf{A}\mathbf{V}_j = \mathbf{V}_j\mathbf{T}_j + \beta_{j+1}v_{j+1}e_j^T, \tag{3.3}$$

kde e_j je j-tý sloupec matice identity. Porovnání algoritmu 3.1 s algoritmem 3.2 dává

$$v_{j+1} = (-1)^j \frac{r_j}{\|r_j\|}$$

a rovněž vztahy mezi rekurenčními koeficienty

$$\begin{aligned} \alpha_{j+1} &= \frac{1}{\gamma_j} + \frac{\delta_{j-1}}{\gamma_{j-1}}, \quad \delta_{-1} \equiv 0,\ \gamma_{-1} \equiv 1, \\ \beta_{j+2} &= \frac{\sqrt{\delta_j}}{\gamma_j}\,. \end{aligned}$$

Konečně, užijeme-li substituci

$$x_j = x_0 + \mathbf{V}_j\, y_j \tag{3.4}$$

a ortogonální vztahy mezi r_j a bází $\{v_1, v_2, \ldots, v_j\}$ prostoru $\mathcal{K}_j(\mathbf{A}, r_0)$, dostáváme

$$\begin{aligned} 0 &= \mathbf{V}_j^T r_j = \mathbf{V}_j^T(b - \mathbf{A}x_j) = \mathbf{V}_j^T(r_0 - \mathbf{A}\mathbf{V}_j y_j) \\ &= e_1\|r_0\| - \mathbf{V}_j^T\mathbf{A}\mathbf{V}_j y_j = e_1\|r_0\| - \mathbf{T}_j y_j\,. \end{aligned}$$

Aproximace řešení x_j metody CG je tudíž určena řešením soustavy

$$\mathbf{T}_j\, y_j = e_1 \|r_0\|\,, \tag{3.5}$$

a následným užitím (3.4).

Ortogonální vztahy tvoří eleganci metody popsané v [32] a reprezentují důležitou vlastnost spojující CG se světem ortogonálních polynomů. Užijeme-li algoritmus 3.1, mohou být j-tá chyba resp. reziduum psány ve formě polynomu v maticové proměnné $\mathbf{A}$ aplikovaného na počáteční chybu resp. reziduum

$$x - x_j = \varphi_j(\mathbf{A})\,(x - x_0), \quad r_j = \varphi_j(\mathbf{A})\, r_0, \quad \varphi_j \in \Pi_j \tag{3.6}$$

kde Π_j označuje třídu polynomů stupně nejvýše j s vlastností $\varphi(0) = 1$ (konstantní člen je roven jedné). Předpokládejme rozklad symetrické matice $\mathbf{A}$ ve tvaru

$$\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T, \quad \mathbf{U}\mathbf{U}^T = \mathbf{U}^T\mathbf{U} = \mathbf{I} \tag{3.7}$$

kde $\mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ a $\mathbf{U} = [u_1, \ldots, u_n]$ je matice, jež má ve svých sloupcích normalizované vlastní vektory matice $\mathbf{A}$. Dosazením (3.7), (3.6) do (3.1) dostáváme

$$\begin{aligned} \|x - x_j\|_{\mathbf{A}} &= \|\varphi_j(\mathbf{A})(x - x_0)\|_{\mathbf{A}} = \min_{\varphi\in\Pi_j} \|\varphi(\mathbf{A})(x - x_0)\|_{\mathbf{A}} = \min_{\varphi\in\Pi_j} \|\varphi(\mathbf{A})r_0\|_{\mathbf{A}^{-1}} \\ &= \min_{\varphi\in\Pi_j} \left\{ \sum_{i=1}^{n} \frac{(r_0, u_i)^2}{\lambda_i}\, \varphi^2(\lambda_i) \right\}^{1/2}. \end{aligned} \tag{3.8}$$

Je zřejmé, že pro symetrickou pozitivně definitní matici $\mathbf{A}$ je rychlost konvergence CG určena rozložením vlastních čísel matice $\mathbf{A}$ a velikostí komponent vektoru r_0 ve směrech jednotlivých vlastních vektorů.

Podobně jako v (3.6), v_{j+1} je určen monickým polynomem ψ_j

$$v_{j+1} = \psi_j(\mathbf{A})\, v_1 \cdot \frac{1}{\beta_2\beta_3 \ldots \beta_{j+1}}\,,$$

který je, jako v (3.8), určen minimalizační podmínkou

$$\|\psi_j(\mathbf{A})v_1\| = \min_{\psi\in\mathcal{M}_j} \|\psi(\mathbf{A})v_1\| = \min_{\psi\in\mathcal{M}_j} \left\{ \sum_{i=1}^{n} (v_1, u_i)^2\, \psi^2(\lambda_i) \right\}^{1/2}, \tag{3.9}$$

kde $\mathcal{M}_j$ označuje třídu monických polynomů stupně j.

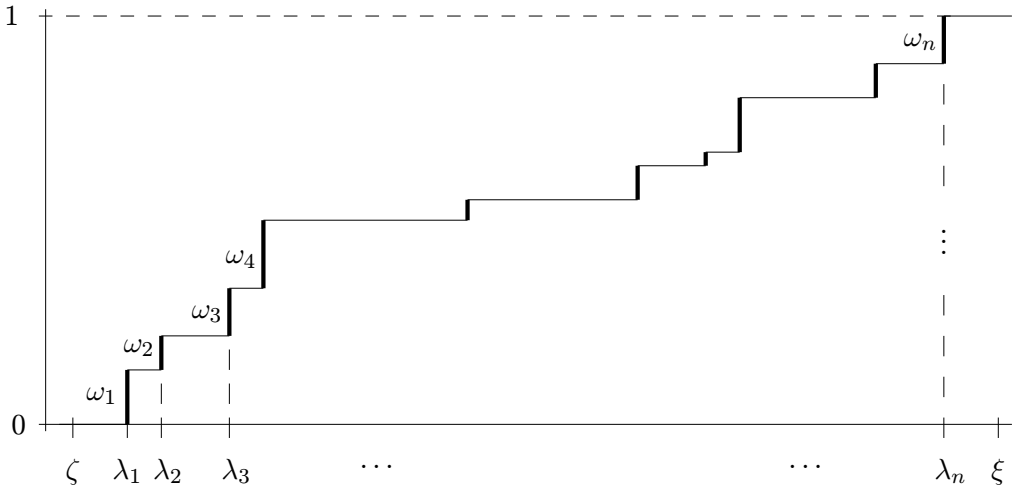
Nyní se pokusíme vysvětlit podstatu algoritmů CG a SL. Vždy, když uvažujeme některý z algoritmů 3.1 (CG) nebo 3.2 (SL), existuje posloupnost $\{1, \psi_1, \ldots, \psi_m\}$, $m = 1, 2, \ldots$ monických ortogonálních polynomů určených (3.9). Tyto polynomy jsou ortogonální vzhledem k diskrétnímu skalárnímu součinu

$$(f, g) = \sum_{i=1}^{n} \omega_i f(\lambda_i) g(\lambda_i)\,, \tag{3.10}$$

kde váhy ω_i jsou určeny vztahem

$$\omega_i = (v_1, u_i)^2, \quad \sum_{i=1}^{n} \omega_i = 1\,. \tag{3.11}$$

Pro jednoduchost označení předpokládáme, že všechna vlastní čísla matice $\mathbf{A}$ jsou různá (pokud má matice $\mathbf{A}$ nějaká násobná vlastní čísla, je rozšíření uvedené konstrukce zřejmé).



Obrázek 3.1: *Distribuční funkce* $\omega(\lambda)$

Nechť jsou vlastní čísla matice $\mathbf{A}$ uspořádána vzestupně a nechť ζ, ξ jsou taková čísla, že $\zeta \le \lambda_1 < \lambda_2 < \cdots < \lambda_n \le \xi$. Uvažujme distribuční funkci $\omega(\lambda)$ s konečným počtem bodů růstu $\lambda_1, \lambda_2, \ldots, \lambda_n$,

$$\begin{aligned} \omega(\lambda) &= 0 && \text{pro} && \lambda < \lambda_1\,, \\ \omega(\lambda) &= \sum_{i=1}^{k} \omega_i && \text{pro} && \lambda_k \le \lambda < \lambda_{k+1}\,, \\ \omega(\lambda) &= 1 && \text{pro} && \lambda_n \le \lambda\,, \end{aligned}$$

viz. obrázek 3.1, a korespondující Riemann-Stieltjesův integrál

$$\int_{\zeta}^{\xi} f(\lambda)\, d\omega(\lambda) = \sum_{i=1}^{n} \omega_i f(\lambda_i)\,. \tag{3.12}$$

Potom může být (3.9) psáno ve tvaru

$$\psi_j = \underset{\psi \in \mathcal{M}_j}{\arg\min} \left\{ \int_{\zeta}^{\xi} \psi^2(\lambda)\, d\omega(\lambda) \right\}^{1/2}, \quad j = 0, 1, 2, \ldots, n\,. \tag{3.13}$$

Ačkoliv je distribuční funkce $\omega(\lambda)$ po částech konstantní funkce s konečným počtem bodů růstu a Riemann-Stieltjesův integrál (3.12) je konečná suma, ukáže se později označení pomocí integrálu velmi užitečné.

Prvních j kroků CG resp. SL s počátečním vektorem $\|r_0\|v_1$ resp. v_1 určuje symetrickou třídiagonální matici $\mathbf{T}_j$ s kladnými mimodiagonálními prvky (3.2). Předpokládejme, analogicky k (3.7), rozklad symetrické matice $\mathbf{T}_j$ ve tvaru

$$\mathbf{T}_j = \mathbf{U}_j \mathbf{\Lambda}_j \mathbf{U}_j^T, \quad \mathbf{U}_j^T \mathbf{U}_j = \mathbf{U}_j \mathbf{U}_j^T = \mathbf{I},$$

kde $\mathbf{\Lambda}_j = \operatorname{diag}(\lambda_1^{(j)}, \ldots, \lambda_j^{(j)})$, $\mathbf{U}_j = [u_1^{(j)}, \ldots, u_j^{(j)}]$. Na matici $\mathbf{T}_j$ se můžeme dívat tak, jako by byla určena j-dimenzionálním CG nebo Lanczosovým procesem pro matici $\mathbf{T}_j$ a vektor $\|r_0\|e_1$ resp. e_1. Zřejmě potom můžeme opakovat konstrukci Riemann-Stieltjesova integrálu popsaného výše pro tento j-dimenzionální proces. Nechť $\zeta \le \lambda_1^{(j)} < \lambda_2^{(j)} < \ldots, \lambda_j^{(j)} \le \xi$ jsou vlastní čísla matice $\mathbf{T}_j$ (Ritzovy hodnoty, musí být různé, viz. např. [46, Kapilola 7]), a

$$\omega_i^{(j)} = (e_1, u_i^{(j)})^2, \quad \sum_{i=1}^{j} \omega_i^{(j)} = 1$$

nechť jsou váhy určeny jako čtverce velikostí komponent vektoru e_1 ve směrech vlastních vektorů matice $\mathbf{T}_j$,

$$\begin{aligned}
&\omega^{(j)}(\lambda) = 0 && \text{pro} && \lambda < \lambda_1^{(j)}\,,\\
&\omega^{(j)}(\lambda) = \sum_{i=1}^{k} \omega_i^{(j)} && \text{pro} && \lambda_k^{(j)} \le \lambda < \lambda_{k+1}^{(j)},\\
&\omega^{(j)}(\lambda) = 1 && \text{pro} && \lambda_j^{(j)} \le \lambda\,.
\end{aligned}$$

Potom je prvních j polynomů z množiny $\{1, \psi_1, \ldots, \psi_n\}$ určených (3.13), rovněž určeno podmínkou založenou na Riemann-Stieltjesově integrálu s distribuční funkcí $\omega^{(j)}(\lambda)$

$$\psi_i = \mathop{\arg\min}_{\psi \in \mathcal{M}_i} \left\{ \int_\zeta^\xi \psi^2(\lambda)\, d\omega^{(j)}(\lambda) \right\}^{1/2}, \quad i = 0, 1, \ldots, j\,.$$

Integrál

$$\int_\zeta^\xi f(\lambda)\, d\omega^{(j)}(\lambda) = \sum_{i=1}^{j} \omega_i^{(j)} f(\lambda_i^{(j)}) \tag{3.14}$$

je dobře známá j-bodová Gaussova kvadratura integrálu (3.12), viz. např. [16]. CG a SL tedy produkují posloupnost distribučních funkcí $\omega^{(1)}(\lambda)$, $\omega^{(2)}(\lambda)$, …, $\omega^{(j)}(\lambda)$, … aproximujících optimálním způsobem (ve smyslu Gaussovy kvadratury) originální distribuční funkci $\omega(\lambda)$, viz. [35], [60, Kapitola XV], [58].

Uvedené skutečnosti jsou poměrně dobře známy. Gaussova kvadratura reprezentuje klasický učebnicový materiál a spojitost mezi CG a Gaussovou kvadraturou byla zdůrazněna již v originální práci [32]. Je tedy nějaký důvod pro opakování těchto skutečností znovu v této práci? Věříme, že pro to máme dobrý důvod. Rádi bychom zdůraznili následující skutečnost. Nejenže CG *souvisí* s Gaussovou kvadraturou (ve smyslu popsaném výše), CG *je Gaussova kvadratura* a tento fakt je klíčový pro pochopení jak matematických vlastností tak chování v konečné aritmetice. Je-li dána matice $\mathbf{A}$ a vektor r_0, jsou integrál (3.12) a jeho aproximace pomocí Gaussovy kvadratury (3.14), $j = 1, 2, \ldots$ jednoznačně určeny. Naopak, distribuční funkce $\omega^{(j)}(\lambda)$ jednoznačně určuje symetrickou třídiagonální matici $\mathbf{T}_j$ a přes (3.5) a (3.4) CG-aproximaci x_j.

Navíc, pro funkci $f(\lambda) = \lambda^{-1}$ dostáváme z (3.8)

$$\|x - x_0\|_{\mathbf{A}}^2 = \|r_0\|^2 \sum_{i=1}^{n} \frac{\omega_i}{\lambda_i} = \|r_0\|^2 \int_\zeta^\xi f(\lambda)\, d\omega(\lambda)\,, \tag{3.15}$$

a užitím (3.3) s $j = n$,

$$\|x - x_0\|_{\mathbf{A}}^2 = (r_0, \mathbf{A}^{-1} r_0) = \|r_0\|^2 (e_1, \mathbf{T}_n^{-1} e_1) \equiv \|r_0\|^2\, (\mathbf{T}_n^{-1})_{11}\,.$$

Konečně, pro $f(\lambda) = \lambda^{-1}$

$$\int_\zeta^\xi f(\lambda)\, d\omega(\lambda) = (\mathbf{T}_n^{-1})_{11}\,. \tag{3.16}$$

Opakováním stejných úvah a postupů při aplikaci CG nebo Lanczosova procesu na $\mathbf{T}_j$ a startovací vektor $\|r_0\| e_1$ dostáváme

$$\int_\zeta^\xi f(\lambda)\, d\omega^{(j)}(\lambda) = (\mathbf{T}_j^{-1})_{11}\,. \tag{3.17}$$

Použití j-bodové Gaussovy kvadratury na (3.12) zapíšeme ve tvaru

$$\int_\zeta^\xi f(\lambda)\, d\omega(\lambda) = \int_\zeta^\xi f(\lambda)\, d\omega^{(j)}(\lambda) + R_j(f) \tag{3.18}$$

kde $R_j(f)$ zastupuje chybu v Gaussově kvadratuře. V dalším odstavci ukážeme několik různých cest, jak lze vyjádřit identitu (3.18).

3.2 Základní vztahy

Našim základním cílem je ukázat souvislost rovnosti (3.18) s hodnotami, jež vystupují v algoritmu CG. Přenásobíme-li (3.18) číslem $\|r_0\|^2$ a užijeme-li volby $f(\lambda) = \lambda^{-1}$, dostáváme

$$\|r_0\|^2 \int_\zeta^\xi \lambda^{-1}\, d\omega(\lambda) \;=\; \|r_0\|^2 \int_\zeta^\xi \lambda^{-1}\, d\omega^{(j)}(\lambda) + \|r_0\|^2 R_j(\lambda^{-1}). \tag{3.19}$$

Užijeme-li (3.15), (3.16) a (3.17), můžeme (3.19) psát ve tvaru

$$\|x - x_0\|_{\mathbf{A}}^2 \;=\; \|r_0\|^2 (\mathbf{T}_n^{-1})_{11} \;=\; \|r_0\|^2 (\mathbf{T}_j^{-1})_{11} + \|r_0\|^2 R_j(\lambda^{-1})\,.$$

V [19, pp. 253-254] bylo dokázáno, že chybu v Gaussově kvadratuře lze vyjádřit ve tvaru

$$R_j(\lambda^{-1}) = \frac{\|x - x_j\|_{\mathbf{A}}^2}{\|r_0\|^2}\,,$$

a proto

$$\|x - x_0\|_{\mathbf{A}}^2 \;=\; \|r_0\|^2 (\mathbf{T}_j^{-1})_{11} + \|x - x_j\|_{\mathbf{A}}^2\,. \tag{3.20}$$

Vztah (3.20) neříká nic jiného, než že hodnota j-té Gaussovy kvadraturní aproximace integrálu (3.16) je doplněk chyby j-té iterace CG procesu měřené pomocí $\|x - x_j\|_{\mathbf{A}}^2 / \|r_0\|^2$. Tento vztah byl odvozen v [8] použitím teorie momentů. Byl tématem dalších navazujících a rozšiřujících prací motivovaných odhadem normy chyby v metodě CG viz. [11], [17] a [19], kde byl užíván plně v kontextu s Gaussovou kvadraturou. Práce v tomto směru pokračovala a vedla na články [18], [38], [39], [7].

Ve všech pracech zmíněných výše byla Gaussova kvadratura vyčíslována pomocí hodnoty $(\mathbf{T}_j^{-1})_{11}$, kterou bylo nutné vypočítat. Zajímavé podoby vztahu (3.20) si povšiml Warnick [66]. Užijeme-li (3.5) a (3.4), dostáváme

$$\begin{aligned}
\|r_0\|^2 (\mathbf{T}_j^{-1})_{11} &= \|r_0\|\, e_1^T\, \mathbf{T}_j^{-1}\, e_1 \|r_0\| \\
&= \|r_0\|\, v_1^T \mathbf{V}_j\, \mathbf{T}_j^{-1} e_1 \|r_0\| = (\|r_0\| v_1)^T \left(\mathbf{V}_j \mathbf{T}_j^{-1} e_1 \|r_0\| \right) \\
&= r_0^T (x_j - x_0)\,,
\end{aligned}$$

a $(\mathbf{T}_j^{-1})_{11}$ je dáno jednoduchým skalárním součinem vektorů, jež jsou k dispozici při výpočtu CG algoritmem. Konečně

$$\|x - x_0\|_{\mathbf{A}}^2 \;=\; r_0^T (x_j - x_0) + \|x - x_j\|_{\mathbf{A}}^2\,. \tag{3.21}$$

Na tento zajímavý vztah jsme byli upozorněni profesorem Saylorem [51]. Nutno poznamenat, že odvození vztahu (3.21) je založeno na ortogonálním vztahu $v_1^T V_j = e_1$, který

v aritmetice s konečnou platí jen vyjímečně. Tato skutečnost, jak objasníme v oddíle 5.2.3, komplikuje výrazně použití vztahu (3.21) v praktických výpočtech.

Vztah matematicky ekvivalentní k (3.21) lze odvodit jednoduchou algebraickou manipulací bez užití Gaussovy kvadratury. Platí

$$\begin{aligned}(3.22)\qquad (x-x_0)^T\mathbf{A}(x-x_0) &= (x-x_j+x_j-x_0)^T\mathbf{A}(x-x_0)\\ &= (x-x_j)^T\mathbf{A}(x-x_0)+(x_j-x_0)^T\mathbf{A}(x-x_0)\\ &= (x-x_j)^T\mathbf{A}(x-x_j+x_j-x_0)+(x_j-x_0)^Tr_0\\ &= \|x-x_j\|_{\mathbf{A}}^2+(x-x_j)^T\mathbf{A}(x_j-x_0)+r_0^T(x_j-x_0)\\ &= \|x-x_j\|_{\mathbf{A}}^2+r_j^T(x_j-x_0)+r_0^T(x_j-x_0)\end{aligned}$$

a tudíž

$$(3.23)\qquad \|x-x_0\|_{\mathbf{A}}^2 = r_0^T(x_j-x_0)+r_j^T(x_j-x_0)+\|x-x_j\|_{\mathbf{A}}^2.$$

Na pravé straně (3.23) se objevil, v porovnání s (3.21), nový výraz $r_j^T(x_j-x_0)$, který je v přesné aritmetice roven nule. Jak bude ukázáno v oddíle 5.2.2, tento výraz bude mít důležitý korekční efekt v aritmetice s konečnou přesností.

Vztahy (3.20), (3.21) a (3.23) reprezentují různé matematicky ekvivalentní podoby vztahu (3.19). Zatímco v (3.20) je j-bodová Gaussova kvadratura vyjádřena jako $(\mathbf{T}_j^{-1})_{11}$, v (3.21) a (3.23) je její hodnota počítána pomocí skalárních součinů vektorů, které jsou k dispozici během iteračního procesu.

Avšak, jak bylo zmíněno v úvodu, existuje daleko jednodušší vyjádření **A**-normy chyby v metodě CG ekvivalentní s (3.19). Je velmi překvapující, že ačkoliv je klíčový vztah uveden v původní práci Hestenese a Stiefela [32, Věta 6.1, vztah (6:2), p. 416], nebyl tento vztah (alespoň podle toho, co je nám známo) nikdy vztažen ke Gaussově kvadratuře a rovněž nebyl nikdy citován. Vzhledem k jeho důležitosti uvádíme jeho odvození. Z definice **A**-normy a z algoritmu 3.1 plyne

$$\begin{aligned}\|x-x_i\|_{\mathbf{A}}^2-\|x-x_{i+1}\|_{\mathbf{A}}^2 &= x_i^T\mathbf{A}x_i-2x_i^Tb-x_{i+1}^T\mathbf{A}x_{i+1}+2x_{i+1}^Tb\\ &= -\gamma_i^2p_i^T\mathbf{A}p_i+2\gamma_ip_i^T(b-\mathbf{A}x_i)\\ &= \gamma_i\|r_i\|^2\,.\end{aligned}$$

Konečně, pro $0\le j<k\le n$, je

$$(3.24)\qquad \|x-x_j\|_{\mathbf{A}}^2-\|x-x_k\|_{\mathbf{A}}^2=\sum_{i=j}^{k-1}\big(\|x-x_i\|_{\mathbf{A}}^2-\|x-x_{i+1}\|_{\mathbf{A}}^2\big)=\sum_{i=j}^{k-1}\gamma_i\|r_i\|^2,$$

a (3.19) lze psát ve tvaru

$$(3.25)\qquad \|x-x_0\|_{\mathbf{A}}^2 = \sum_{i=0}^{j-1}\gamma_i\|r_i\|^2+\|x-x_j\|_{\mathbf{A}}^2.$$

Součet výrazů $\gamma_i\|r_i\|^2$ lze jednoduše určit z algoritmu metody CG; činitelé γ_i a $\|r_i\|^2$ jsou k dispozici v každém iteračním kroku. Nutno podotknout, že při odvozování (3.24) jsme použili pouze lokální ortogonality mezi po sobě jdoucími rezidui a směrovými vektory a vztahu mezi rekurzivním a skutečným reziduem. Vyhnuli jsme se použití vzájemné ortogonality mezi vektory s obecně různými indexy. Tato skutečnost bude velmi podstatná při zkoumání vlivu zaokrouhlovacích chyb na vztah (3.24) v konečné aritmetice, jak uvidíme v oddíle 5.2.1.

Poznamenejme, že (3.25) lze odvodit i jiným způsobem, viz. např. [4]. Uvědomíme-li si, že platí

$$x_j = x_0 + \sum_{i=0}^{j-1} \gamma_i p_i, \quad x = x_0 + \sum_{i=0}^{n-1} \gamma_i p_i,$$

můžeme na základě **A**-ortogonality mezi směrovými vektory vyjádřit kvadrát **A**-normy j-té chyby ve tvaru

$$\|x - x_j\|_{\mathbf{A}}^2 = \sum_{i=j}^{n-1} \gamma_i^2 p_i^T \mathbf{A} p_i = \sum_{i=j}^{n-1} \gamma_i \|r_i\|^2.$$

Vyjádříme-li analogickým způsobem i kvadrát **A**-normy počáteční chyby, odvodíme jednoduchou algebraickou manipulací vztah (3.25). Uvedeným postupem jsme chtěli poukázat na skutečnost, že samotné použití globální ortogonality resp. **A**-ortogonality při odvozování formule nemusí nutně vést k tomu, že výsledný vztah v konečné aritmetice neplatí. Nevhodná volba postupu při odvození formule však vždy komplikuje její analýzu v konečné aritmetice.

3.3 Odhad A-normy chyby

Naším cílem je v každém iteračním kroku j odhadnout **A**-normu chyby $\|x - x_j\|_{\mathbf{A}}$. Použijeme-li myšlenku publikovanou v [19, pp. 28-29], může být vztahu (3.20) využito při odhadování **A**-normy chyby v CG následujícím způsobem. Užijeme-li (3.20) a předpokladu $\|x - x_0\|_{\mathbf{A}}^2 = \|r_0\|^2 (\mathbf{T}_n^{-1})_{11}$, dostáváme

$$\|x - x_j\|_{\mathbf{A}}^2 \;=\; \|r_0\|^2 \left[(\mathbf{T}_n^{-1})_{11} - (\mathbf{T}_j^{-1})_{11} \right] .$$

V [19] bylo navrženo nahradit neznámou hodnotu $(\mathbf{T}_n^{-1})_{11}$ již vypočtenou $(\mathbf{T}_k^{-1})_{11}$ pro nějaké k, $k > j$. Uvedený nápad však nebyl v [19] plně rozvinut a jeho použití v aritmetice s konečnou přesností bylo pouze omezené. Numericky vhodný postup výpočtu rozdílu $(\mathbf{T}_k^{-1})_{11} - (\mathbf{T}_j^{-1})_{11}$ byl navržen až v práci [18]. K uvedenému problému se vrátíme v oddíle 5.1.

Nyní zobecníme myšlenky uvedené v [19] a [18]. Odečteme-li vztahy (3.19) pro čísla iterací j a $j+d$, kde d je přirozené číslo, eliminujeme neznámý integrál (3.16) a dostaneme

$$\|r_0\|^2 R_j(\lambda^{-1}) = \|r_0\|^2 \left(\int_\zeta^\xi \lambda^{-1} \, d\omega^{(j+d)}(\lambda) - \int_\zeta^\xi \lambda^{-1} \, d\omega^{(j)}(\lambda) \right) + \|r_0\|^2 R_{j+d}(\lambda^{-1}).$$

Podobně užitím (3.20), (3.21), (3.23) a (3.25) dostaneme

$$\|x - x_j\|_{\mathbf{A}}^2 \;=\; \|r_0\|^2 \left[(\mathbf{T}_{j+d}^{-1})_{11} - (\mathbf{T}_j^{-1})_{11} \right] + \|x - x_{j+d}\|_{\mathbf{A}}^2 , \tag{3.26}$$

$$\|x - x_j\|_{\mathbf{A}}^2 \;=\; r_0^T (x_{j+d} - x_j) + \|x - x_{j+d}\|_{\mathbf{A}}^2 , \tag{3.27}$$

$$\begin{aligned} \|x - x_j\|_{\mathbf{A}}^2 \;=\; & r_0^T (x_{j+d} - x_j) - r_j^T (x_j - x_0) + r_{j+d}^T (x_{j+d} - x_0) \\ & + \; \|x - x_{j+d}\|_{\mathbf{A}}^2 \end{aligned} \tag{3.28}$$

a

$$\|x - x_j\|_{\mathbf{A}}^2 \;=\; \sum_{i=j}^{j+d-1} \gamma_i \|r_i\|^2 + \|x - x_{j+d}\|_{\mathbf{A}}^2 . \tag{3.29}$$

Připomeňme nyní, že pro symetrickou pozitivně definitní matici **A** je **A**-norma chyby ostře klesající. Je-li d voleno tak, že

$$\|x - x_j\|_{\mathbf{A}}^2 \gg \|x - x_{j+d}\|_{\mathbf{A}}^2 , \tag{3.30}$$

potom zanedbáním $\|x - x_{j+d}\|_{\mathbf{A}}^2$ na pravé straně rovnic (3.26), (3.27), (3.28) a (3.29) získáme dolní odhad **A**-normy chyby v j-tém kroku, který za předpokladu (3.30) rozumně aproximuje **A**-normu chyby. Označíme-li

$$\eta_{j,d} = \|r_0\|^2 \, [(\mathbf{T}_{j+d}^{-1})_{11} - (\mathbf{T}_j^{-1})_{11}], \tag{3.31}$$

$$\mu_{j,d} = r_0^T (x_{j+d} - x_j), \tag{3.32}$$

$$\vartheta_{j,d} = r_0^T (x_{j+d} - x_j) - r_j^T (x_j - x_0) + r_{j+d}^T (x_{j+d} - x_0) \tag{3.33}$$

a

$$\nu_{j,d} = \sum_{i=j}^{j+d-1} \gamma_i \|r_i\|^2, \tag{3.34}$$

získáme odhady $\eta_{j,d}$, $\mu_{j,d}$, $\vartheta_{j,d}$ a $\nu_{j,d}$ druhé mocniny **A**-normy j-té chyby, jež jsou skrze (3.19) spojeny s Gaussovou kvadraturou. Jak je vidět, odhady $\mu_{j,d}$, $\vartheta_{j,d}$ a $\nu_{j,d}$ jsou jednoduše počitatelné. K jejich vyjádření potřebujeme hodnoty z následujících d iterací a zřejmě tedy v iteračním kroku $j + d$ získáme aproximaci **A**-normy chyby z kroku j.

Ze vztahů (3.26), (3.27), (3.28) a (3.29) je zřejmé, že v přesné aritmetice platí

$$\eta_{j,d} = \mu_{j,d} = \vartheta_{j,d} = \nu_{j,d} . \tag{3.35}$$

V aritmetice s konečnou přesností se však ztrácí ortogonalita resp. **A**-ortogonalita a dokonce i lineární nezávislost mezi počítanými rezidui resp. směrovými vektory. Důsledkem toho nebude vztah (3.35) v konečné aritmetice platit a různé odhady budou dávat různé informace o konvergenci. Velmi důležitý je rovněž fakt, že hodnoty počítané pomocí (3.19)–(3.23) a (3.26)–(3.29) se mohou zásadně lišit od svých protějšků počítaných v přesné aritmetice a je tedy na místě otázka, zda *mají výše uvedené vztahy nějaký význam pro hodnoty počítané v aritmetice s konečnou přesností*. Každá práce pojednávající o tomto tématu by se měla vážně zabývat uvedenou otázkou a jejím zodpovězením prokázat, že uvedené výsledky a vztahy jsou korektní i v aritmetice s konečnou přesností. Pro vztahy (3.20), (3.26) a $\eta_{j,d}$ je odpověď (s jistým omezením) uvedena v [19]. Dříve než vysvětlíme a rozšíříme analýzu zaokrouhlovacích chyb na ostatní odhady, připomeneme stručně v následující kapitole základní výsledky týkající se chování CG a SL v aritmetice s konečnou přesností a podrobně popíšeme a odhadneme zaokrouhlovací chyby vznikající při aplikaci algoritmu CG v konečné aritmetice.

4

CG v konečné aritmetice

Nasazení CG *na počítačích s konečnou aritmetikou s sebou přineslo řadu problémů. K jejich řešení bylo nezbytné pochopit chování a matematický model* CG *v konečné aritmetice. Popíšeme, jak matematický model* CG *v konečné aritmetice vznikal, vysvětlíme jeho základní myšlenku a zformulujeme princip zpoždění. Formálně popíšeme chyby vznikající při výpočtu koeficientů a vektorů algoritmu* CG *v konečné aritmetice a odhadneme velikosti těchto zaokrouhlovacích chyb. Budeme se zabývat problémem zachování lokální ortogonality. Poznatky z této kapitoly využijeme v kapitole 5 při analýze odhadů* $\mathbf{A}$*-normy chyby v konečné aritmetice. Závěr kapitoly patří numerických experimentům ve kterých ukazujeme, že odhady vznikajících zaokrouhlovacích chyb jsou často velmi nadhodnocené a zabýváme se možností jejich zeslabení.*

Hestenes a Stiefel [32] si byli dobře vědomi vlivu zaokrouhlovacích chyb. Sami píší v [32, str. 441], že zaokrouhlovací chyby se neprojeví jen za velmi neobvyklých okolností a že aproximace x_n, která by v přesné aritmetice byla řešením, je pouze jeho aproximací, jež může být dále vylepšována pokračováním algoritmu. Uvažovali CG jako *iterační* metodu. Zdůraznění odpovídá rozporu s obecně a často prezentovaným názorem, že metoda CG byla původně chápána jako metoda finitní a její iterační charakter byl „objeven“ až v [47] (viz. diskuse v [47]).

Je nutné si uvědomit, že algoritmy metod vytvořených v exaktním matematickém světě mohou v konečné aritmetice úplně změnit své chování a počítat hodnoty, které nesouvisí se svými ideálními vzory. Skutečnost, že výpočty algoritmu neztrácí svoje základní vlastnosti a že se na ně můžeme spolehnout i v konečné aritmetice je vždy nutné dokázat, což znamená, že je potřeba vytvořit matematický model výpočtů algoritmu v konečné aritmetice, jak se to povedlo na základě prací [22], [28] u metody sdružených gradientů. Práce vlastně říkají, že metoda CG zůstává metodou CG i v konečné aritmetice, je jen trošku deformována a dochází ke zpoždění, nikoliv ztrátě, konvergence. V přirovnání bychom celou situaci mohli chápat tak, že jdeme po ledě, občas nám to podklouzne, tudíž uděláme více kroků než kdybychom šli po pevné zemi, ale vždy dojdeme ke stejnému cíli. Tím jsme stručně vysvětlili problém CG versus konečná aritmetika, který v této kapitole na základě prací [22], [28], [57], [59] popíšeme podrobněji.

Při naší další diskusi potřebujeme rozlišit, kdy hovoříme o hodnotách vypočtených algoritmem CG v konečné aritmetice a kdy o ideálních (přesných) hodnotách určených algoritmem CG v přesné aritmetice. Abychom zjednodušili diskusi, zaveďme následující označení. Pod pojmem *ideální* CG budeme rozumnět algoritmus CG v přesné aritmetice. Hodnoty určené ideální CG budeme označovat jako *ideální* nebo *přesné*. Pod označením (FP)CG budeme rozumnět algoritmus CG aplikovaný v konečné aritmetice (finite precision arithmetic) a o hodnotách určených tímto algoritmem budeme hovořit jako o hodnotách *vypočtených*.

Kapitola má následující strukturu. V sekci 4.1 vysvětlujeme matematický model výpočtů metody CG v konečné aritmetice a v následující sekci 4.2 diskutujeme o zpoždění konvergence způsobené zaokrouhlovacími chybami. Na základě standardního modelu arit-

metiky s pohyblivou řádovou čárkou odhadujeme v 4.3 zaokrouhlovací chyby vznikající při výpočtu algoritmu v jednotlivých rekurencích. V sekci 4.4 se zabýváme lokální ortogonalitou mezi reziduem a směrovým vektorem z následující iterace a výsledek formulujeme ve větě 4.1 (bude využit v kapitole 5). V 4.5 následují numerické experimenty, které ukazují, jaké zaokrouhlovací chyby vznikají při reálných výpočtech algoritmu CG.

4.1 Matematický model CG v konečné aritmetice

Matematický exaktní svět je vhodný k vytváření nových myšlenek, postupů a metod. Po jejich případné aplikaci v praxi (např. při počítání v konečné aritmetice počítače) se může stát, že výpočet s použitím daného algoritmu se od svého ideálního vzoru v mnoha ohledech liší a je ho nutné opět matematickými prostředky popsat, vytvořit jeho matematický model.

Přeneseme-li algoritmus CG resp. SL do prostředí konečné aritmetiky, zjistíme, že ortogonalita mezi počítanými vektory $\{v_1, \ldots, v_j\}$ se obvykle velmi rychle vytrácí, dochází dokonce k jejich lineární závislosti a následkem toho ke ztrátě finitnosti metod. Po dlouhou dobu se zdálo, že zaokrouhlovací chyby zničí všechny dobré vlastnosti jimiž metody vynikají. Navzdory tomu však obě dávaly rozumné výsledky.

První krok na dlouhé cestě k pochopení chování CG v konečné aritmetice provedl Paige, který se ve své disertační práci [41] zabýval otázkou ztráty ortogonality. Dokázal, že ztráta ortogonality je možná pouze ve směrech konvergujících Ritzových vektorů $z_i^{(j)}$. Více detailů je možné nalézt v [41], [42], [43], [44]. Jeho práce vytvořily základ, na kterém se dalo dále stavět, mohl začít vznikat matematický model chování CG v konečné aritmetice.

Podstatný krok v tomto směru vykonala Greenbaum ve své práci [22]. Abychom zkrátili naši diskusi, zaměřme se pouze na symetrické pozitivně definitní matice $\mathbf{A}$ a na metodu sdružených gradientů (detaily obsahující informace o symetrickém Lanczosově algoritmu mohou být nalezeny v [22] a [28]). Předpokládejme použití CG k řešení systému lineárních rovnic $\mathbf{A}x = b$ na počítači, který používá aritmetiku s konečnou přesností reprezentovanou strojovou přesností ε. V [22] je ukázáno, že pro zvolené j jsou $\mathbf{A}$-normy chyby $\|x - x_i\|_{\mathbf{A}}$ v krocích 1 až j *velmi blízké* $\overline{\mathbf{A}}$-normám chyb $\|\overline{x} - \overline{x}_i\|_{\overline{\mathbf{A}}}$, jež jsou určeny ideální CG aplikovanou na jistý symetrický pozitivně definitní systém $\overline{\mathbf{A}}(j)\overline{x}(j) = \overline{b}(j)$. Tento systém a počáteční aproximace $\overline{x}_0(j)$ závisí na iteračním kroku j (pro jednoduchost budeme vyjadřovat tuto závislost pouze bude-li to nutné). Matice $\overline{\mathbf{A}}$ je větší než matice $\mathbf{A}$. Její vlastní čísla musí ležet ve *velmi malých* intervalech okolo vlastních čísel matice $\mathbf{A}$ a, jak bylo dokázáno v [56], pro každé vlastní číslo matice $\mathbf{A}$ musí existovat alespoň jedno blízké vlastní číslo matice $\overline{\mathbf{A}}$. Navíc pro každé vlastní číslo λ_i matice $\mathbf{A}$, $i = 1, \ldots, n$ (podobně jako v oddílu 3.1 předpokládáme, bez újmy na obecnosti, že všechna vlastní čísla matice $\mathbf{A}$ jsou různá) jsou váhy $\omega_i = (v_1, u_i)^2$ rovny součtu vah příslušných k vlastním číslům matice $\overline{\mathbf{A}}$, která jsou soustředěna v blízkosti λ_i.

Užijme terminologie Gaussovy kvadratury, viz. oddíl 3.1 a 3.2. Vztahy (3.15)–(3.20) ukazují, že konvergence ideální CG, měřená $\mathbf{A}$-normou chyby, je určena Gaussovou kvadraturou pro Riemann-Stieltjesův integrál

$$\int_{\zeta}^{\xi} \lambda^{-1} \, d\omega(\lambda), \tag{4.1}$$

s monotónní po částech konstantní distribuční funkcí $\omega(\lambda)$. Funkce $\omega(\lambda)$ má n bodů růstu $\lambda_1, \ldots, \lambda_n$ (vlastní čísla matice $\mathbf{A}$) s velikostí skoků $\omega_1, \ldots, \omega_n$ (čtverce velikostí komponent v_1 v bázi vlastních vektorů, viz. obrázek 3.1). Distribuční funkce $\overline{\omega}(\lambda)$ určená $\overline{\mathbf{A}}$, $\overline{b}$ a $\overline{x}_0$ (příslušná k j krokům (FP)CG) má mnoho bodů růstu, jež jsou umístěny v okolí

vlastních čísel původní matice. Tato distribuční funkce aproximuje původní distribuční funkci $\omega(\lambda)$ určenou pomocí $\mathbf{A}$, b a x_0. Výsledky Gaussovy kvadratury aplikované na Riemann-Stieltjesův integrál

$$\int_{\zeta}^{\xi} \lambda^{-1} \, d\overline{\omega}(\lambda),$$

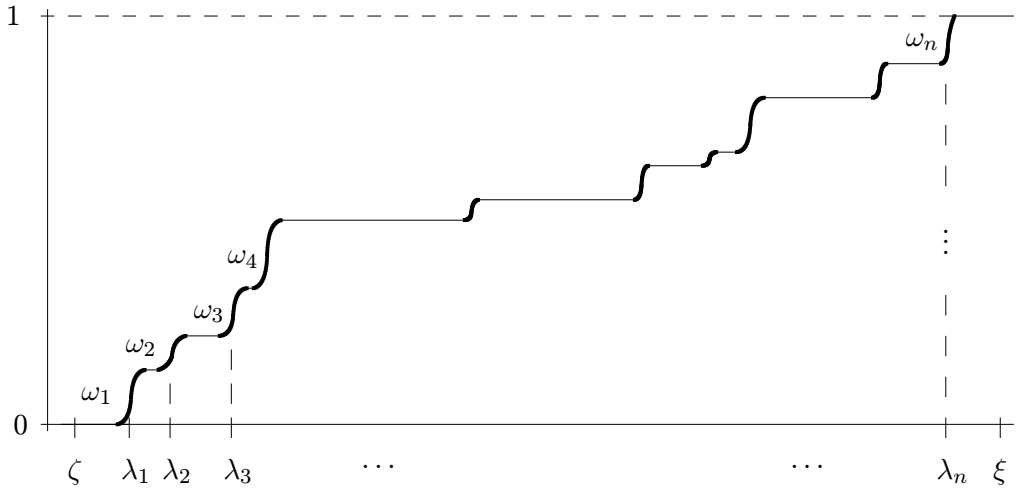
potom určují, *s případnou malou nepřesností*, konvergenci (FP)CG aplikované na $\mathbf{A}x = b$.

Dříve než budeme pokračovat dále, musíme komentovat pojmy „*velmi blízké*“, „*velmi malých*“ a „*s případnou malou nepřesností*“, které byly užity výše. Formulace vztahů mezi $\mathbf{A}$, b, x_0 a $\overline{\mathbf{A}}$, $\overline{b}$, $\overline{x}_0$ v článcích [22], [56] a [19] obsahuje výrazy, jež jsou vztaženy různým komplikovaným způsobem ke strojové přesnosti ε. Aktuální velikost výrazů, jež se vyskytují v uvedených pracech není důležitá; spíše dokumentuje obecné technické problémy vznikající při analýze zaokrouhlovacích chyb způsobené manipulací s komplikovanými výrazy než že by říkala něco o přesnosti popsaných vztahů. Podstatný je základní pohled, který byl (často velmi slabými) odhady zaokrouhlovacích chyb vytvořen. Na výpočty (FP)CG pro systém $\mathbf{A}x = b$ lze (s jistou malou nepřesností) hledět jako na výpočty ideální CG pro určitý systém $\overline{\mathbf{A}}\overline{x} = \overline{b}$. Numerické experimenty ukazují, že přesnost této relace je daleko větší, než technicky komplikované teoretické výpočty v [22] dokazují.

Analýza zaokrouhlovacích chyb v CG ve smyslu zpětné chyby (založená na transformaci zaokrouhlovacích chyb v jednotlivých iteracích v modifikaci originálních vstupních dat), kterou jsme dosud prezentovali, může být shrnuta následujícím způsobem. Pro výpočty z j kroků (FP)CG aplikované na systém $\mathbf{A}x = b$ existují jisté $\overline{\mathbf{A}}(j)$, $\overline{x}(j)$, $\overline{b}(j)$ a $\overline{x}_0(j)$, určené pro konkrétní zaokrouhlovací chyby (objevující se v prvním až j-tém kroku) s vlastnostmi popsanými výše. Závislost $\overline{\mathbf{A}}$, $\overline{x}$, $\overline{b}$ a $\overline{x}_0$ na j a na konkrétních zaokrouhlovacích chybách omezuje další aplikaci tohoto pojetí. Pro $k > j$ nejenže se korespondující $\overline{\mathbf{A}}(k)$, $\overline{x}(k)$, $\overline{b}(k)$ a $\overline{x}_0(k)$ liší od $\overline{\mathbf{A}}(j)$, $\overline{x}(j)$, $\overline{b}(j)$ a $\overline{x}_0(j)$, ale dokonce mohou mít podstatně různé dimenze.

Pro překonání této obtíže, bylo v [28] navrženo zkonstruovat pro daný systém $\mathbf{A}x = b$ a počáteční přiblížení x_0 daleko rozsáhlejší systém $\hat{\mathbf{A}}\hat{x} = \hat{b}$ s počáteční aproximací $\hat{x}_0$, kde matice $\hat{\mathbf{A}}$ má velké množství vlastních čísel v malých intervalech, jejichž středem jsou původní vlastní čísla. Navíc požadujeme, aby součet vah určených $\hat{\mathbf{A}}$, $\hat{b}$ a $\hat{x}_0$ korespondujících k i-tému intervalu byl roven váze ω_i určené pomocí (3.11). Velikost těchto intervalů by měla být úměrná $\varepsilon\|\mathbf{A}\|$; konstanta úměrnosti nehraje důležitou roli. Počet vlastních čísel matice $\hat{\mathbf{A}}$ příslušných k jednotlivým vlastním číslům matice $\mathbf{A}$ rovněž není důležitý za předpokladu, že jejich počet je větší než možný počet Ritzových kopií, které mohou aproximovat každé individuální vlastní číslo matice $\mathbf{A}$ při výpočtech (FP)CG. Uvažujeme-li počet iterací (FP)CG aplikované na $\mathbf{A}x = b$ omezený číslem m, potom každý ze shluků vlastních čísel matice $\hat{\mathbf{A}}$ může být například zvolen tak, že obsahuje m vlastních čísel, což dává celkovou dimenzi $\hat{\mathbf{A}}$ rovnu $n * m$.

Toto zobecnění může být zdůvodněno následujícím způsobem. V [22] bylo dokázáno, že ideální rekurence pro matici, jejíž vlastní čísla jsou shromážděna v malých intervalech obsahujících původní vlastní čísla λ_i, $i = 1, \ldots, n$, mohou být považovány za lehce pozměněné rekurence pro matici $\mathbf{A}$ (a korespondující b, x_0). Při výpočtech v aritmetice s konečnou přesností je toto pozměnění výsledkem vlivu zaokrouhlovacích chyb. Důkladné vyšetřování (Grcar [21], Simon [53], [52]) ukázalo, že ztráta ortogonality (jež může být považována za primární důsledek elementárních zaokrouhlovacích chyb a která určuje všechny další rozdíly mezi pozměněnými a ideálními rekurencemi) je v podstatě nezávislá na přesnosti elementárních poruch v rekurencích CG v jednotlivých iteračních krocích. S uvážením uvedených argumentů může být ideální CG aplikovaná na konkrétně zkonstruovaný systém $\overline{\mathbf{A}}(j)\overline{x}(j) = \overline{b}(j)$ považována (s případnou malou nepřesností) za ideální CG aplikovanou na „univerzální“ systém $\hat{\mathbf{A}}\hat{x} = \hat{b}$ (a obě mohou být považovány za lehce pozměněné rekurence ideální CG pro $\mathbf{A}x = b$ s nevýznamnými rozdíly mezi korespon-



Obrázek 4.1: *Distribuční funkce* $\hat{\omega}(\lambda)$

dujícími individuálními poruchami). Tudíž ideální CG pro „univerzální“ systém $\hat{\mathbf{A}}\hat{x} = \hat{b}$ modeluje (FP)CG aplikovanou na systém $\mathbf{A}x = b$.

Shrneme-li vše, (FP)CG může být v podstatě považována za ideální CG aplikovanou na modifikovaný problém $\hat{\mathbf{A}}\hat{x} = \hat{b}$, pro který je konvergence určena Riemann-Stieltjesovým integrálem

$$\int_{\zeta}^{\xi} \lambda^{-1} \, d\hat{\omega}(\lambda), \tag{4.2}$$

s distribuční funkcí $\hat{\omega}(\lambda)$, kterou získáme z $\omega(\lambda)$ *rozmazáním* jednotlivých bodů λ_i, $i = 1, \ldots, n$ do (pokud možno nekonečně) mnoha bodů růstu, jež se nacházejí v těsné blízkosti každého λ_i, přičemž celková velikost přírůstku v okolí λ_i je rovna ω_i (zůstává zachována). Každá $\hat{\omega}(\lambda)$ splňující tyto dvě podmínky dává podobné výsledky. Z pochopitelných důvodů předpokládáme, že matice $\mathbf{A}$ není blízká numericky singulární matici, t.j. $0 < \varepsilon\|\mathbf{A}\| \ll \zeta$ (potom je $\kappa(\mathbf{A}) = \lambda_n/\lambda_1 < \|\mathbf{A}\|/\zeta \ll \frac{1}{\varepsilon}$). Funkce λ^{-1} je hladkou monotónní funkcí na intervalu $\langle\zeta, \xi\rangle$ a tudíž výsledky Gaussovy kvadratury pro Riemann-Stieltjesův integrál s distribučními funkcemi, jež jsou vzájemně velmi blízké (splňující podmínky pro $\hat{\omega}(\lambda)$) musí rovněž být velmi blízké. Je důležité poznamenat, že předchozí výrok platí obecně za předpokladu, že počet uzlů v Gaussově kvadratuře není větší než počet bodů růstu rozmazané distribuční funkce v každém z intervalů. Tato podmínka je triviální a bez újmy na obecnosti předpokládáme, že je splněna. Uvedený argument reprezentuje další zdůvodnění pro nahrazení konkrétního systému $\overline{\mathbf{A}}(j)\overline{x}(j) = \overline{b}(j)$ „univerzálním“ systémem $\hat{\mathbf{A}}\hat{x} = \hat{b}$. V důsledku toho můžeme považovat distribuční funkci $\hat{\omega}(\lambda)$ z (4.2) za *spojitou funkci* dobře aproximující originální skokovou funkci $\omega(\lambda)$, viz. obrázek 4.1.

Pojetí v [28] je neobvyklé. Výpočetní vlastnosti metody CG aplikované na daný systém $\mathbf{A}x = b$ v konečné aritmetice charakterizované strojovou přesností ε jsou modelovány pomocí klasické matematické teorie Gaussovy kvadratury pro určitý Riemann-Stieltjesův integrál. Hodnoty vypovídající o konvergenci CG metody v aritmetice s konečnou přesností získáme jako ideální matematické výsledky Gaussovy kvadratury.

4.2 Zpoždění konvergence

Je dobře známo, že v důsledku ztráty ortogonality je v konečné aritmetice konvergence CG metody zpožděna. Nejasný princip zpoždění konvergence může být nyní formalizován

a matematicky zdůvodněn.

V přesné aritmetice je rychlost konvergence CG aplikované na systém $\mathbf{A}x = b$ dána vztahem (viz. (3.20), (3.17))

$$\|x - x_0\|_{\mathbf{A}}^2 - \|x - x_j\|_{\mathbf{A}}^2 \;=\; \|r_0\|^2 \sum_{i=1}^{j} \frac{\omega_i^{(j)}}{\lambda_i^{(j)}} \qquad \text{(přesná aritmetika)}\,, \tag{4.3}$$

kde pravá strana reprezentuje výsledek j-té Gaussovy kvadratury pro Riemann-Stieltjesův integrál (4.1) násobený $\|r_0\|^2$. V konečné aritmetice nám náš matematický model dává (pouze s malou nepřesností) pokles $\mathbf{A}$-normy chyby pro aktuálně počítanou aproximaci x_j jako výsledek ideální j-bodové Gaussovy kvadratury pro (4.2),

$$\|x - x_0\|_{\mathbf{A}}^2 - \|x - x_j\|_{\mathbf{A}}^2 \;=\; \|r_0\|^2 \sum_{i=1}^{j} \frac{\hat{\omega}_i^{(j)}}{\hat{\lambda}_i^{(j)}} + \mathcal{O}(\varepsilon) \qquad \text{(konečná aritmetika)}\,, \tag{4.4}$$

kde $\mathcal{O}(\varepsilon)$ reprezentuje malé číslo úměrné strojové přesnosti ε.

Výsledek (4.4) Gaussovy kvadratury aplikované na (4.2) může být velmi rozdílný od výsledku (4.3) Gaussovy kvadratury aplikované na (4.1). To můžeme nahlédnout následujícím způsobem. Uvažujme rozmazanou distribuční funkci $\hat{\omega}(\lambda)$. Potom může j-bodová Gaussova kvadratura umístit několik kořenů korespondujících ortogonálních polynomů (viz. (3.10)) blízko nějakého vlastního čísla matice $\mathbf{A}$ (tento jev je způsoben faktem, že existuje interval růstu $\hat{\omega}(\lambda)$ okolo každého λ_i). Ekvivalentně, při výpočtech v konečné aritmetice může několik kopií Ritzových hodnot aproximovat jedno vlastní číslo λ_i a každé další Ritzovo číslo aproximující již dobře aproximované vlastní číslo vždy znamená nějakou další nadbytečnou informaci v aproximaci vlastního vektoru a případnou další numerickou ztrátu lineární nezávislosti mezi Lanczosovými vektory $v_1, v_2, \ldots$. Rozložení původních vlastních čísel $\lambda_1, \ldots, \lambda_n$ určuje, skrze rozmazanou distribuční funkci $\hat{\omega}(\lambda)$, frekvenci vytváření nadbytečných kopií jednotlivých vlastních čísel a ztrátu numerické dimenze počítaných Krylovových prostorů. Vynecháme podrobnosti technického charakteru, viz. [58]. Abychom dostali podobný výsledek jako j-bodová Gaussova kvadratura pro původní integrál (4.1), může Gaussova kvadratura pro (4.2) s rozmazanou distribuční funkcí $\hat{\omega}(\lambda)$ potřebovat více než j-uzlů, jelikož některé z uzlů jsou *vyplýtvány* při vytváření přebytečných kopií.

Předpokládejme výsledek (4.3) j-bodové Gaussovy kvadratury pro (4.1). Aby platilo

$$\sum_{i=1}^{m} \frac{\hat{\omega}_i^{(m)}}{\hat{\lambda}_i^{(m)}} \;\simeq\; \sum_{i=1}^{j} \frac{\omega_i^{(j)}}{\lambda_i^{(j)}}\,, \tag{4.5}$$

pro nějaké m, potřebuje Gaussova kvadratura pro (4.2) alespoň $m = j + \Delta(j)$ kroků, kde $\Delta(j)$ je počet uzlů umístěných v blízkosti již aproximovaných vlastních čísel. Naopak, je-li dáno m, může být korespondující hodnota j přibližně určena jako (numerický) řád matice $n \times m$, jejíž sloupce jsou tvořeny počítanými aproximacemi vlastních vektorů matice $\mathbf{A}$ (a ten je přibližně roven numerickému řádu matice počítaných Lanczosových vektorů $\mathbf{V}_m$). Výsledkem našich úvah je princip zpoždění: *Zpoždění konvergence* (FP)CG *je určeno rozdílem mezi číslem iterace m a numerickým řádem matice počítaných Lanczosových vektorů $\mathbf{V}_m$. Jinými slovy, úrovně přesnosti dosažené v m krocích* (FP)CG *by bylo dosaženo v j krocích přesné* CG*, kde j je rovno (numerickému) řádu počítané matice $\mathbf{V}_m$.* Ačkoliv některé technické detaily stále čekají na objasnění (viz. [58]), je uvedený model dobře ospravedlněn teoretickými výsledky a potvrzen numerickými experimenty.

Na obrázcích 4.2 a 4.3 demonstrujeme numericky chování CG v konečné aritmetice a princip zpoždění. Na obrázku 4.2 znázorňujeme výsledky z počítání (FP)CG pro matici $\mathbf{A} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T$, kde $\mathbf{Q}$ je ortogonální matice, která vznikla z QR-rozkladu náhodně generované matice (počítané příkazem `randn(n)` v systému MATLAB), a $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ je diagonální matice s vlastními čísly λ_i počítanými podle předpisu

$$\lambda_i = \lambda_1 + \frac{i-1}{n-1}(\lambda_n - \lambda_1)\,\rho^{n-i}, \quad i = 2, \ldots, n-1, \tag{4.6}$$

viz. [56]. Zvolili jsme $n = 48$, $\lambda_1 = 0.1$, $\lambda_n = 1000$, $\rho = 0.9$, $x = (1, \ldots, 1)^T$, $b = \mathbf{A}x$ a $x_0 = (0, \ldots, 0)^T$. Hodnoty počítané ideální CG jsme modelovali pomocí CG v konečné aritmetice s dvojnásobně reortogonalizovanými reziduovými vektory (viz. [28]). Tyto hodnoty značíme pomocí (E). Na obrázku 4.2 vidíme, že čerchovaná křivka $\mathbf{A}$-normy chyby (E) CG může být velmi odlišná od křivky $\mathbf{A}$-normy chyby v (FP)CG, zakreslené plnou čarou. Ke ztrátě ortogonality mezi (FP) Lanczosovými vektory (tečkovaná křivka) měřené Frobeniovou normou $\|\mathbf{I} - \mathbf{V}_j^T\mathbf{V}_j\|_F$ dochází po pár iteracích. Ztráta ortogonality mezi Lanczosovými vektory, jež jsou počítány CG algoritmem s dvojnásobně reortogonalizovanými reziduovými vektory je zakreslena tečkami. Zůstává, jak je dobře vidět, na úrovni blízké strojové přesnosti. Experimenty byly provedeny v programovém systému MATLAB 5.1 na osobním počítači se strojovou přesností $\varepsilon \approxeq 2.2 \times 10^{-16}$.

Pro demonstraci principu zpoždění jsme počítali pro každý krok m (FP)CG numerický řád $j(m)$ počítané matice $\mathbf{V}_m$ (podrobněji, $j(m)$ je zde určeno jako počet singulárních hodnot matice $\mathbf{V}_m$, které se nacházejí pod určitou hranicí, řekněme 0.1, což indikuje, že $\mathbf{V}_m$ nemá více než $j(m)$ sloupců, které jsou silně lineárně nezávislé). Potom jsme posunuli bod na (FP) konvergenční křivce *horizontálně* z iterace m do iterace $j(m)$. Výsledky jsou ukázány pro $\mathbf{A}$-normu chyby (čárkované křivky), euklidovskou normu chyby (plné křivky) a relativní reziduovou normu (tečkované křivky) na obrázku 4.3. Porovnání čerchované křivky na obrázku 4.2 s posunutou čárkovanou křivkou na obrázku 4.3 ukazuje téměř úplnou shodu. Podobnou shodu je rovněž možno vidět na ostatních konvergenčních charakteristikách (korespondující křivky na obrázku 4.2 jsme neznázorňovali).

4.3 Předpoklady a odhady zaokrouhlovacích chyb

Podobně jako v oddíle 4.1 uvažujeme, že matice $\mathbf{A}$ je dostatečně vzdálená od matic numericky singulárních, t.j.

$$\kappa(\mathbf{A}) \ll \varepsilon^{-1}. \tag{4.7}$$

Připomeňme, že matice $\mathbf{A}$ je symetrická a pozitivně definitní a proto platí

$$\|\mathbf{A}\| = \lambda_{max}, \qquad \|\mathbf{A}\|^{1/2} = \lambda_{max}^{1/2} = \|\mathbf{A}^{1/2}\|$$

a podobně

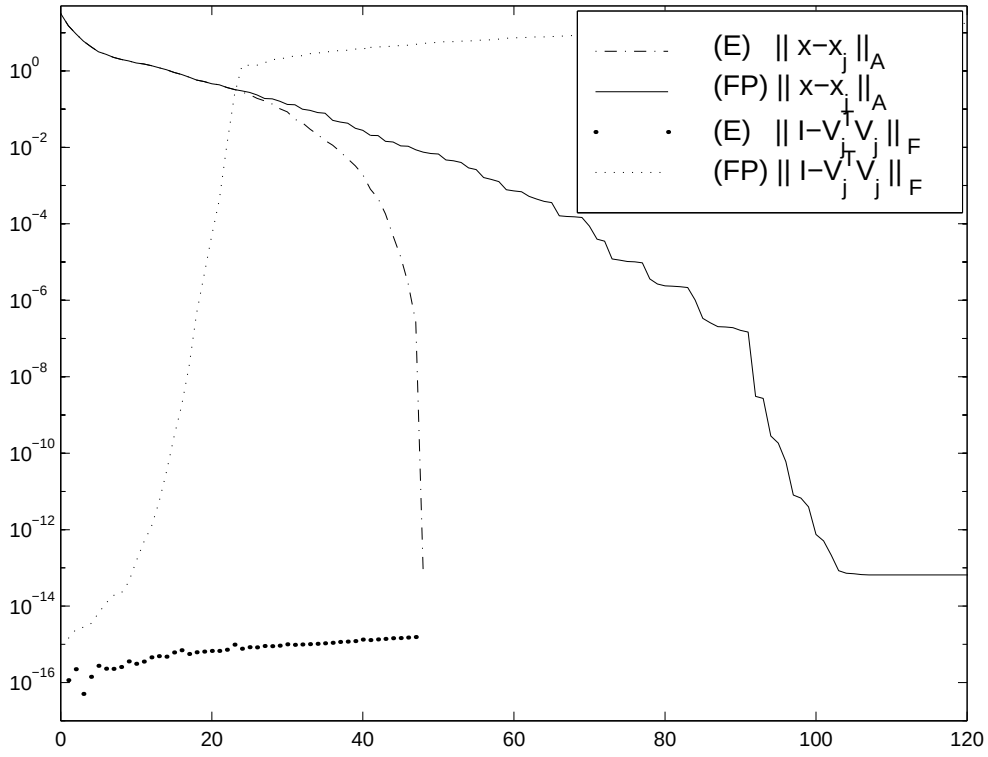
$$\|\mathbf{A}^{-1}\| = \lambda_{min}^{-1}, \qquad \|\mathbf{A}^{-1}\|^{1/2} = \lambda_{min}^{-1/2} = \|\mathbf{A}^{-1/2}\|,$$

kde λ_{min} a λ_{max} jsou nejmenší a největší vlastní číslo matice $\mathbf{A}$. $\mathbf{A}$-normu chyby můžeme vyjádřit ve tvaru

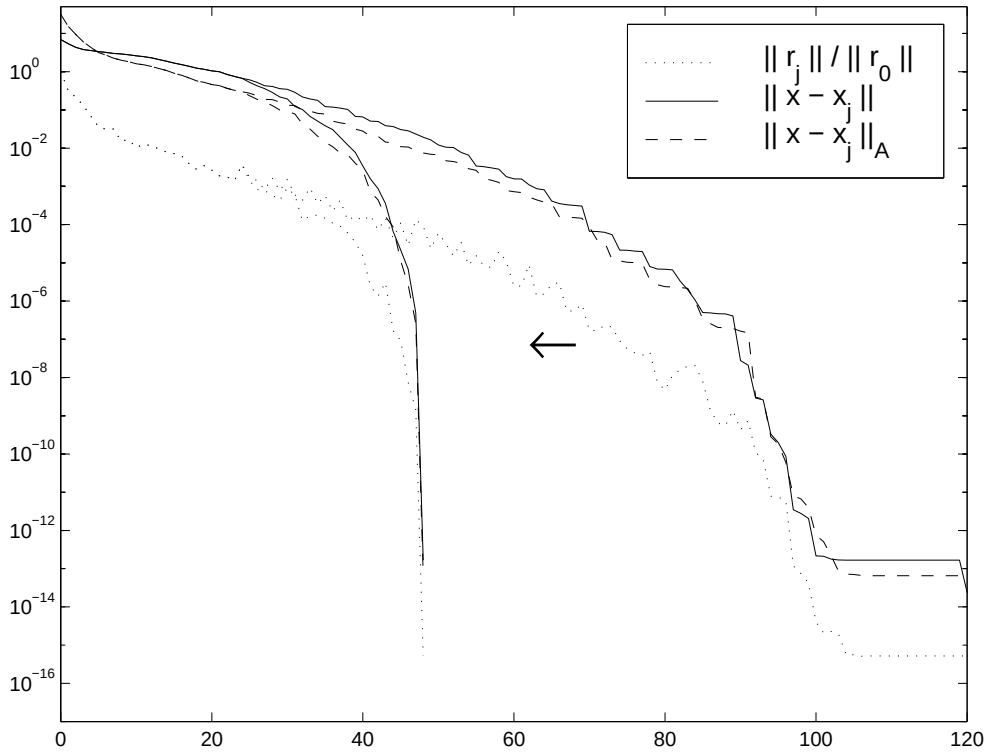
$$\|x - x_j\|_{\mathbf{A}} = \|\mathbf{A}^{1/2}(x - x_j)\|.$$

V následujícím textu zvolíme způsob popisu CG v konečné aritmetice a budeme se věnovat odhadu velikosti zaokrouhlovacích chyb, které při výpočtech vznikají.

Model aritmetiky s pohyblivou řádovou čárkou. Pro jednoduchost a přehlednost našich úvah budeme při závislosti na strojové přesnosti ε uvažovat pouze výrazy závislé



Obrázek 4.2: Ideální CG a (FP)CG



Obrázek 4.3: Princip zpoždění

lineárně. Členy úměrné vyšším mocninám ε budeme zahrnovat do výrazů $\mathcal{O}(\varepsilon^2)$. Předpokládáme následující standardní model aritmetiky s pohyblivou řádovou čárkou, viz. např. [33, (2.4)], reprezentovanou strojovou přesností ε

$$\mathrm{fl}[a \circ b] = (a \circ b)(1+\delta), \qquad |\delta| \le \varepsilon. \tag{4.8}$$

Symbol $\circ$ zastupuje operace sčítání, násobení a dělení, a a b jsou čísla v uvažované konečné aritmetice a $\mathrm{fl}[a]$ označuje, že výpočet a je prováděn v konečné aritmetice. Na základě tohoto modelu platí pro vektory v a w a pro číslo α tyto standardní výsledky (viz. [20] nebo [23])

$$\|\alpha\, v - \mathrm{fl}[\alpha\, v]\| \le \varepsilon\, \|\alpha\, v\|, \tag{4.9}$$

$$\|v + w - \mathrm{fl}[v+w]\| \le \varepsilon\, (\|v\| + \|w\|), \tag{4.10}$$

$$|(v,w) - \mathrm{fl}[(v,w)]| \le n\,(\varepsilon + \mathcal{O}(\varepsilon^2))\, \|v\|\, \|w\|. \tag{4.11}$$

Násobíme-li vektor p_j maticí $\mathbf{A}$ v konečné aritmetice, vzniká chyba, kterou lze vyjádřit následujícím vztahem

$$\mathrm{fl}[\mathbf{A}p_j] = \mathbf{A}p_j + \varepsilon\, \widetilde{\mathbf{A}}_j p_j, \tag{4.12}$$

kde $\tilde{\mathbf{A}}_j$ je matice $n \times n$ a velikost jejích prvků $\widetilde{a}_{ik}^{(j)}$ je ohraničena velikostí prvků a_{ik} původní matice $\mathbf{A}$, platí

$$|\widetilde{a}_{ik}^{(j)}| \le |a_{ik}|.$$

Normu chyby $\varepsilon\, \widetilde{\mathbf{A}}_j p_j$ lze odhadnout následovně

$$\|\varepsilon\, \widetilde{\mathbf{A}}_j p_j\| \le \varepsilon\, c\, \|\mathbf{A}\|\, \|p_j\|.$$

Je-li $\mathbf{A}$ matice s nejvýše m nenulovými prvky na řádku a je-li součin matice-vektor počítán standardním způsobem, je

$$c = m\, n^{1/2}.$$

Lokální chyby v rekurencích. Při výpočtech v aritmetice s konečnou přesností lze hodnoty počítané metodou CG popsat následujícím způsobem

$$\gamma_j = \frac{(r_j, r_j)}{(p_j, \mathbf{A}p_j)} + \varepsilon\, \zeta_j^{\gamma}, \tag{4.13}$$

$$x_{j+1} = x_j + \gamma_j\, p_j + \varepsilon\, l_j^x, \tag{4.14}$$

$$r_{j+1} = r_j - \gamma_j\, \mathbf{A}p_j + \varepsilon\, l_j^r, \tag{4.15}$$

$$\delta_j = \frac{(r_{j+1}, r_{j+1})}{(r_j, r_j)} + \varepsilon\, \zeta_j^{\delta}, \tag{4.16}$$

$$p_{j+1} = r_{j+1} + \delta_j\, p_j + \varepsilon\, l_j^p. \tag{4.17}$$

Zde $\varepsilon\, \zeta_j^{\gamma}$ a $\varepsilon\, \zeta_j^{\delta}$ značí lokální chyby výpočtu koeficientů γ_j a δ_j (skaláry) a $\varepsilon\, l_j^x$, $\varepsilon\, l_j^r$ a $\varepsilon\, l_j^p$ lokální chyby výpočtu vektorů x_{j+1}, r_{j+1} a p_{j+1} (vektory).

Monotonie A-normy chyby v konečné aritmetice. Na základě výsledků z [22] a [28] platí, že monotonie $\mathbf{A}$-normy chyby a euklidovské normy chyby zůstává zachována rovněž v konečné aritmetice (až na případnou malou chybu).

Rozdíl mezi rekurzivním a skutečným reziduem. V algoritmu metody CG je rekurzivně počítáno reziduum r_k, které je ideálně rovno $b - \mathbf{A}x_k$. V (FP)CG tato rovnost

obecně neplatí. Velmi často se stává, že norma rekurzivně počítaného rezidua eventuálně konverguje k nule, zatímco norma skutečného rezidua $b - \mathbf{A}x_k$ stagnuje na jisté hladině přesnosti a tudíž se obě rezidua mohou značně lišit. Označíme-li rozdíl mezi oběma rezidui následujícím způsobem

$$\varepsilon\, f_j \equiv r_j - (b - \mathbf{A}x_j), \tag{4.18}$$

platí na základě prací [54] a [23]

$$\|\varepsilon\, f_j\| \leq \varepsilon\, \|\mathbf{A}\| \Theta_j\, \mathcal{O}(j\, c) + \mathcal{O}(\varepsilon^2), \tag{4.19}$$

kde

$$\Theta_j \equiv \|x\| + \max_{i \leq j}\{\|x_i\|\}. \tag{4.20}$$

Přenásobíme-li vztah (4.14) pro vektor x_{j+1} maticí $-\mathbf{A}$ a přičteme-li k obou stranám vektor b, dostáváme

$$b - \mathbf{A}x_{j+1} = b - \mathbf{A}x_j - \gamma_j\, \mathbf{A}p_j - \varepsilon\, \mathbf{A}l_j^x.$$

Odečteme-li právě uvedenou rovnost od vztahu (4.15) pro vektor r_{j+1}, dostáváme na základě definice (4.18)

$$f_{j+1} = f_j + l_j^r + \mathbf{A}l_j^x \tag{4.21}$$

a užitím indukce

$$f_{j+1} = f_0 + \sum_{i=0}^{j} l_i^r + \mathbf{A} \sum_{i=0}^{j} l_i^x. \tag{4.22}$$

Aplikaci našich výsledků budeme uvažovat pouze do té doby (iterace), dokud reziduum $b - \mathbf{A}x_j$ nedosáhne limitní hladiny přesnosti, t.j. platí-li

$$\|b - \mathbf{A}x_j\| \gg \varepsilon\, \|f_j\|. \tag{4.23}$$

Potom již údaje počítané (FP)CG, z nichž se konstruují odhady, nevypovídají nic o blízkosti počítané aproximace k přesnému řešení. Definujeme-li číslo $\varepsilon\, F_j$ vztahem

$$\|r_j\| = \|b - \mathbf{A}x_j\|\, (1 + \varepsilon\, F_j), \tag{4.24}$$

zřejmě platí

$$|\varepsilon\, F_j| = \frac{|\|r_j\| - \|b - \mathbf{A}x_j\||}{\|b - \mathbf{A}x_j\|} \leq \frac{\|r_j - (b - \mathbf{A}x_j)\|}{\|b - \mathbf{A}x_j\|} = \frac{\varepsilon\, \|f_j\|}{\|b - \mathbf{A}x_j\|}$$

a $\varepsilon\, F_j$ je podle (4.23) a (4.24) numericky malá hodnota, $\varepsilon\, F_j \ll 1$. Poznamenejme, že velikost hodnoty $\varepsilon\, F_j$ je úměrná řádové vzdálenosti normy skutečného rezidua a limitní hladiny přesnosti skutečného rezidua.

Protože je $b - \mathbf{A}x_j = \mathbf{A}(x - x_j)$, plyne zřejmě z (4.24) vztah

$$\lambda_{min}^{1/2}(\mathbf{A})\, \|x - x_j\|_{\mathbf{A}}(1 + \varepsilon\, F_j) \leq \|r_j\| \leq \lambda_{max}^{1/2}(\mathbf{A})\, \|x - x_j\|_{\mathbf{A}}(1 + \varepsilon\, F_j). \tag{4.25}$$

Odhady velikosti lokálních chyb. Podle pravidel (4.8)–(4.11) standardního modelu aritmetiky s pohyblivou řádovou čárkou lze lokální chyby $\varepsilon\, l_j^x$, $\varepsilon\, l_j^r$ a $\varepsilon\, l_j^p$ vznikající při výpočtu vektorů x_{j+1}, r_{j+1} a p_{j+1} odhadnout (jako např. v [23]) následovně:

$$\varepsilon\,\|l_j^x\| \;\le\; \varepsilon\,\{\|x_j\| + 2\,\|\gamma_j p_j\|\} + \mathcal{O}(\varepsilon^2), \tag{4.26}$$

$$\varepsilon\,\|l_j^r\| \;\le\; \varepsilon\,\{\|r_j\| + 2\,\|\gamma_j \mathbf{A} p_j\| + c\,\|\mathbf{A}\|\|\gamma_j p_j\|\} + \mathcal{O}(\varepsilon^2), \tag{4.27}$$

$$\varepsilon\,\|l_j^p\| \;\le\; \varepsilon\,\{\|r_{j+1}\| + 2\,\|\delta_j p_j\|\} + \mathcal{O}(\varepsilon^2). \tag{4.28}$$

Ukažme na základě (4.27) vztah mezi normou vektoru $\varepsilon\, l_j^r$ a $\mathbf{A}$-normou j-té chyby, který využijeme později při analýze zaokrouhlovacích chyb. Odhadněme jednotlivé výrazy na pravé straně (4.27). Pro první výraz platí

$$\varepsilon\,\|r_j\| = \varepsilon\,\|\mathbf{A}(x - x_j)\|(1 + \varepsilon\, F_j) \le \varepsilon\,\|\mathbf{A}\|^{1/2}\|x - x_j\|_{\mathbf{A}}(1 + \varepsilon\, F_j).$$

Při odhadování druhého výrazu na pravé straně (4.27) použijeme vztah (4.15), předpoklad (4.24) a monotonii $\mathbf{A}$-normy chyby, dostáváme

$$\begin{aligned}
2\varepsilon\,\|\gamma_j \mathbf{A} p_j\| &= \varepsilon\, 2\,\|r_{j+1} - r_j\| + \mathcal{O}(\varepsilon^2) \\
&= \varepsilon\, 2\,\|\mathbf{A}^{1/2}\mathbf{A}^{1/2}(x_j - x_{j+1})\|(1 + \varepsilon\, F_j) + \mathcal{O}(\varepsilon^2) \\
&\le \varepsilon\, 2\,\|\mathbf{A}\|^{1/2}\|\mathbf{A}^{1/2}(x_j - x + x - x_{j+1})\|(1 + \varepsilon\, F_j) + \mathcal{O}(\varepsilon^2) \\
&\le \varepsilon\, 4\,\|\mathbf{A}\|^{1/2}\|x - x_j\|_{\mathbf{A}}(1 + \varepsilon\, F_j) + \mathcal{O}(\varepsilon^2)
\end{aligned}$$

a obdobným způsobem odhadneme i třetí výraz

$$\begin{aligned}
\varepsilon\, c\,\|\mathbf{A}\|\|\gamma_j p_j\| &= \varepsilon\, c\,\|\mathbf{A}\|\|x_{j+1} - x_j\| + \mathcal{O}(\varepsilon^2) \\
&= \varepsilon\, c\,\|\mathbf{A}\|\|\mathbf{A}^{-1/2}\mathbf{A}^{1/2}(x_j - x_{j+1})\| + \mathcal{O}(\varepsilon^2) \\
&\le \varepsilon\, c\,\|\mathbf{A}\|^{1/2}\kappa(\mathbf{A})^{1/2}\|x_j - x + x - x_{j+1}\|_{\mathbf{A}} + \mathcal{O}(\varepsilon^2) \\
&\le \varepsilon\, c\,\|\mathbf{A}\|^{1/2}\kappa(\mathbf{A})^{1/2}\|x - x_j\|_{\mathbf{A}} + \mathcal{O}(\varepsilon^2).
\end{aligned}$$

Na základě uvedených nerovností platí

$$\varepsilon\,\|l_j^r\| \le \varepsilon\,\|\mathbf{A}\|^{1/2}\kappa(\mathbf{A})^{1/2}\|x - x_j\|_{\mathbf{A}}\,\mathcal{O}(c) + \mathcal{O}(\varepsilon^2). \tag{4.29}$$

Věnujme se nyní odhadu velikosti chyb $\varepsilon\,\zeta_j^\gamma$ a $\varepsilon\,\zeta_j^\delta$, vznikajících při výpočtu koeficientů γ_j a δ_j. V konečné aritmetice platí (za předpokladu $n\,\varepsilon \ll 1$)

$$\delta_j = \frac{\|r_{j+1}\|^2(1 + n\,\mathcal{O}(\varepsilon))}{\|r_j\|^2(1 + n\,\mathcal{O}(\varepsilon))}(1 + \mathcal{O}(\varepsilon)) = \frac{\|r_{j+1}\|^2}{\|r_j\|^2}(1 + n\,\mathcal{O}(\varepsilon)) + \mathcal{O}(\varepsilon^2) \tag{4.30}$$

a proto je

$$\varepsilon\,|\zeta_j^\delta| \le \varepsilon\,\frac{\|r_{j+1}\|^2}{\|r_j\|^2}\,\mathcal{O}(n) + \mathcal{O}(\varepsilon^2). \tag{4.31}$$

Pro výpočet jmenovatele koeficientu γ_j platí

$$\begin{aligned}
\mathrm{fl}[(p_j, \mathrm{fl}[\mathbf{A}p_j])] &= (p_j, \mathbf{A}p_j) + n\,\|\mathbf{A}p_j\|\|p_j\|\mathcal{O}(\varepsilon) + c\,\|\mathbf{A}\|\|p_j\|^2\mathcal{O}(\varepsilon) + \mathcal{O}(\varepsilon^2) \\
&= (p_j, \mathbf{A}p_j)\left(1 + n\,\frac{\|\mathbf{A}p_j\|\|p_j\|}{\|p_j\|_{\mathbf{A}}^2}\mathcal{O}(\varepsilon) + c\,\frac{\|\mathbf{A}\|\|p_j\|^2}{(p_j, \mathbf{A}p_j)}\mathcal{O}(\varepsilon)\right) + \mathcal{O}(\varepsilon^2) \\
&= (p_j, \mathbf{A}p_j)\left(1 + \left[n\,\|\mathbf{A}\|^{1/2}\frac{\|p_j\|_{\mathbf{A}}\|p_j\|}{\|p_j\|_{\mathbf{A}}^2} + c\,\frac{\|\mathbf{A}\|\|p_j\|^2}{\lambda_{min}\|p_j\|^2}\right]\mathcal{O}(\varepsilon)\right) + \mathcal{O}(\varepsilon^2) \\
&= (p_j, \mathbf{A}p_j)\big(1 + n\,\kappa(\mathbf{A})^{1/2}\mathcal{O}(\varepsilon) + c\,\kappa(\mathbf{A})\mathcal{O}(\varepsilon)\big) + \mathcal{O}(\varepsilon^2) \\
&= (p_j, \mathbf{A}p_j)\big(1 + (c + n)\,\kappa(\mathbf{A})\mathcal{O}(\varepsilon)\big) + \mathcal{O}(\varepsilon^2)
\end{aligned}$$

a celkem dostáváme (za předpokladu $(c+n)\,\kappa(\mathbf{A})\,\varepsilon \ll 1$)

$$\begin{aligned}\gamma_j &= \frac{\|r_j\|^2(1+n\,\mathcal{O}(\varepsilon))}{(p_j,\mathbf{A}p_j)(1+(c+n)\,\kappa(\mathbf{A})\mathcal{O}(\varepsilon))+\mathcal{O}(\varepsilon^2)}(1+\mathcal{O}(\varepsilon))\\ &= \frac{\|r_j\|^2}{(p_j,\mathbf{A}p_j)}(1+(c+n)\,\kappa(\mathbf{A})\mathcal{O}(\varepsilon))+\mathcal{O}(\varepsilon^2).\end{aligned}$$

Velikost chyby vznikající při výpočtu γ_j lze odhadnout následovně:

$$\varepsilon\,|\zeta_j^\gamma| \le \varepsilon\,\kappa(\mathbf{A})\frac{\|r_j\|^2}{(p_j,\mathbf{A}p_j)}\,\mathcal{O}(c+n)+\mathcal{O}(\varepsilon^2). \tag{4.32}$$

Pro jemnější manipulaci s lokálními chybami $\varepsilon\,\zeta_j^\gamma$ a $\varepsilon\,l_j^r$ odvodíme vztahy, které více vypovídají o struktuře těchto lokálních zaokrouhlovacích chyb. S využitím (4.12), (4.15) a (4.13) lze chyby $\varepsilon\,l_j^r$ a $\varepsilon\,\zeta_j^\gamma$ psát ve tvaru

$$\varepsilon\,l_j^r = -\varepsilon\,\gamma_j\widetilde{\mathbf{A}}_j p_j + \varepsilon\,\widetilde{l}_j^r, \tag{4.33}$$

$$\varepsilon\,\zeta_j^\gamma = -\varepsilon\,\frac{\|r_j\|^2}{(p_j,\mathbf{A}p_j)}\frac{(p_j,\widetilde{\mathbf{A}}_j p_j)}{(p_j,\mathbf{A}p_j)} + \varepsilon\,\widetilde{\zeta}_j^\gamma, \tag{4.34}$$

kde

$$\varepsilon\,\|\widetilde{l}_j^r\| \le \varepsilon\,\{\|r_j\| + 2\,\|\gamma_j\mathbf{A}p_j\|\} + \mathcal{O}(\varepsilon^2), \tag{4.35}$$

$$\varepsilon\,|\widetilde{\zeta}_j^\gamma| \le \varepsilon\,\frac{\|r_j\|^2}{(p_j,\mathbf{A}p_j)}\kappa(\mathbf{A})^{1/2}\,\mathcal{O}(n)+\mathcal{O}(\varepsilon^2). \tag{4.36}$$

Vztah (4.33) s odhadem (4.35) lze lehce odvodit rozepsáním rekurence pro vektor r_{j+1} v konečné aritmetice

$$r_{j+1} = \mathrm{fl}[r_j - \mathrm{fl}[\gamma_j\mathrm{fl}[\mathbf{A}p_j]]],$$

dosazením za $\mathrm{fl}[\mathbf{A}p_j]$ podle (4.12) a užitím standardních pravidel pro odhad velikosti zaokrouhlovacích chyb (4.9)–(4.11).

Odvození vztahu (4.34) a odhadu (4.36) je komplikovanější a proto jej nyní uvádíme. Zřejmě platí

$$\begin{aligned}\mathrm{fl}[(p_j,\mathrm{fl}[\mathbf{A}p_j])] &= \mathrm{fl}[(p_j,\mathbf{A}p_j+\varepsilon\,\widetilde{\mathbf{A}}_j p_j)]\\ &= (p_j,\mathbf{A}p_j)+\varepsilon\,(p_j,\widetilde{\mathbf{A}}_j p_j)+n\,\|\mathbf{A}p_j\|\|p_j\|\mathcal{O}(\varepsilon)+\mathcal{O}(\varepsilon^2),\\ &= (p_j,\mathbf{A}p_j)\left(1+\varepsilon\,\frac{(p_j,\widetilde{\mathbf{A}}_j p_j)}{(p_j,\mathbf{A}p_j)}+n\,\|\mathbf{A}\|^{1/2}\frac{\|p_j\|_{\mathbf{A}}\|p_j\|}{\|p_j\|_{\mathbf{A}}^2}\mathcal{O}(\varepsilon)\right)+\mathcal{O}(\varepsilon^2)\\ &= (p_j,\mathbf{A}p_j)\left(1+\varepsilon\,\frac{(p_j,\widetilde{\mathbf{A}}_j p_j)}{(p_j,\mathbf{A}p_j)}+n\,\kappa(\mathbf{A})^{1/2}\mathcal{O}(\varepsilon)\right)+\mathcal{O}(\varepsilon^2).\end{aligned}$$

Koeficient γ_j počítaný v konečné aritmetice lze potom zapsat ve tvaru (za předpokladu $(c+n)\,\kappa(\mathbf{A})\,\varepsilon \ll 1$)

$$\begin{aligned}\gamma_j &= \frac{\|r_j\|^2}{(p_j,\mathbf{A}p_j)}\,\frac{1+n\,\mathcal{O}(\varepsilon)}{1+\varepsilon\,\frac{(p_j,\widetilde{\mathbf{A}}_j p_j)}{(p_j,\mathbf{A}p_j)}+n\,\kappa(\mathbf{A})^{1/2}\mathcal{O}(\varepsilon)}\,(1+\mathcal{O}(\varepsilon))\\ &= \frac{\|r_j\|^2}{(p_j,\mathbf{A}p_j)}\left(1-\varepsilon\,\frac{(p_j,\widetilde{\mathbf{A}}_j p_j)}{(p_j,\mathbf{A}p_j)}+n\,\kappa(\mathbf{A})^{1/2}\mathcal{O}(\varepsilon)+\mathcal{O}(\varepsilon^2)\right)+\mathcal{O}(\varepsilon^2)\\ &= \frac{\|r_j\|^2}{(p_j,\mathbf{A}p_j)}-\varepsilon\,\frac{\|r_j\|^2}{(p_j,\mathbf{A}p_j)}\frac{(p_j,\widetilde{\mathbf{A}}_j p_j)}{(p_j,\mathbf{A}p_j)}+n\,\frac{\|r_j\|^2}{(p_j,\mathbf{A}p_j)}\kappa(\mathbf{A})^{1/2}\mathcal{O}(\varepsilon)+\mathcal{O}(\varepsilon^2)\end{aligned} \tag{4.37}$$

a z (4.37) plyne (4.34) a (4.36).

4.4 Lokální ortogonalita

Je dobře známo, že v tříkrokových lanczosových rekurencích je lokální ortogonalita zachována na úrovni strojového přesnosti ε (viz. [43]). V tomto odstavci odvodíme analogii výše uvedeného výsledku pro (FP)CG. Budeme se zabývat velikostí skalárního součinu vektorů p_j a r_{j+1}, $j = 1, 2, \ldots$, jež jsou v ideální CG ortogonální. Výsledků zde odvozených použijeme v analýze zaokrouhlovacích chyb u odhadu založeného na vztahu (3.29). Problém lokální ortogonality souvisí s vlastnostmi algoritmu CG v konečné aritmetice a ne pouze s analýzou odhadů **A**-normy chyby a proto se jeho zkoumání věnujeme v této sekci. Náš výsledek jsme shrnuli v následující větě.

Věta 4.1 *Uvažujme algoritmus* CG *aplikovaný v aritmetice s konečnou přesností. Chyby vznikající při výpočtu nechť jsou charakterizovány vztahy* (4.13)–(4.17) *a nechť platí* $(c + n)\,\kappa(\mathbf{A})\,\varepsilon \ll 1$. *Potom pro počítané vektory* p_j *a* r_{j+1} *platí*

$$|p_j^T r_{j+1}| \leq \varepsilon\,\|r_j\|^2 \kappa(\mathbf{A})^{1/2}\,\mathcal{O}(j\,n + j^2/2) + \mathcal{O}(\varepsilon^2). \tag{4.38}$$

Důkaz. Pro $i = 0$ dostáváme

$$\begin{aligned} p_0^T r_1 &= r_0^T (r_0 - \gamma_0 \mathbf{A} r_0 + \varepsilon\, l_0^r) \\ &= \|r_0\|^2 - \left(\frac{\|r_0\|^2}{r_0^T \mathbf{A} r_0} + \varepsilon\,\zeta_0^\gamma\right) r_0^T \mathbf{A} r_0 + \varepsilon\, r_0^T l_0^r \\ &= \varepsilon\, r_0^T (l_0^r - \zeta_0^\gamma \mathbf{A} r_0). \end{aligned}$$

Definujeme-li $M_0 \equiv r_0^T (l_0^r - \zeta_0^\gamma \mathbf{A} r_0)$, je

$$p_0^T r_1 = \varepsilon\, M_0. \tag{4.39}$$

Násobením vztahu (4.15) vektorem p_j^T získáme rovnost

$$\begin{aligned} p_j^T r_{j+1} &= p_j^T r_j - \gamma_j p_j^T \mathbf{A} p_j + \varepsilon\, p_j^T l_j^r \\ &= (r_j + \delta_{j-1} p_{j-1} + \varepsilon\, l_{j-1}^p)^T r_j - \left(\frac{\|r_j\|^2}{p_j^T \mathbf{A} p_j} + \varepsilon\,\zeta_j^\gamma\right) p_j^T \mathbf{A} p_j + \varepsilon\, p_j^T l_j^r. \end{aligned}$$

Roznásobíme-li závorky, dostáváme po odečtení hodnoty $\|r_j\|^2$ vztah

$$p_j^T r_{j+1} = \delta_{j-1}\, p_{j-1}^T r_j + \varepsilon\,\{r_j^T l_{j-1}^p - \zeta_j^\gamma p_j^T \mathbf{A} p_j + p_j^T l_j^r\}. \tag{4.40}$$

Označíme-li výraz ve složené závorce symbolem M_j,

$$M_j \equiv r_j^T l_{j-1}^p - \zeta_j^\gamma p_j^T \mathbf{A} p_j + p_j^T l_j^r, \tag{4.41}$$

lze (4.40) vyjádřit ve tvaru

$$p_j^T r_{j+1} = \delta_{j-1}\, p_{j-1}^T r_j + \varepsilon\, M_j.$$

Opakování stejného kroku pro $p_{j-1}^T r_j, \ldots, p_1^T r_2$ vede za použití (4.39) na rovnost

$$p_j^T r_{j+1} = \varepsilon \sum_{i=0}^{j} M_i \left(\prod_{k=i}^{j-1} \delta_k\right). \tag{4.42}$$

Dosadíme-li do (4.42) za δ_k podle (4.30) výraz

$$\frac{\|r_{k+1}\|^2}{\|r_k\|^2}(1 + n\,\mathcal{O}(\varepsilon)) + \mathcal{O}(\varepsilon^2),$$

dostáváme

$$\begin{aligned} p_j^T r_{j+1} &= \varepsilon \sum_{i=0}^{j} M_i \prod_{k=i}^{j-1} \Big(\frac{\|r_{k+1}\|^2}{\|r_k\|^2}(1 + n\,\mathcal{O}(\varepsilon)) + \mathcal{O}(\varepsilon^2) \Big) \\ &= \varepsilon \sum_{i=0}^{j} M_i \prod_{k=i}^{j-1} \frac{\|r_{k+1}\|^2}{\|r_k\|^2} + \mathcal{O}(\varepsilon^2) \\ &= \varepsilon \, \|r_j\|^2 \sum_{i=0}^{j} \frac{M_i}{\|r_i\|^2} + \mathcal{O}(\varepsilon^2). \qquad (4.43) \end{aligned}$$

Zbývá vyřešit problém odhadu výrazů $\varepsilon\,|M_i|/\|r_i\|^2$. Dříve než tak učiníme, vyjádříme výraz M_i za použití (4.33) a (4.34) v jiné podobě. Platí

$$\begin{aligned} \varepsilon\, M_i &= \varepsilon\, r_i^T l_{i-1}^p - \varepsilon\, \zeta_i^\gamma p_i^T \mathbf{A} p_i + \varepsilon\, p_i^T l_i^r \\ &= \varepsilon\, r_i^T l_{i-1}^p + \Big(\varepsilon\, \frac{\|r_i\|^2}{p_i^T \mathbf{A} p_i} \frac{p_i^T \widetilde{\mathbf{A}}_i p_i}{p_i^T \mathbf{A} p_i} - \varepsilon\, \widetilde{\zeta}_i^\gamma \Big) p_i^T \mathbf{A} p_i + p_i^T \big(- \varepsilon\, \gamma_i \widetilde{\mathbf{A}}_i p_i + \varepsilon\, \widetilde{l}_i^r \big) \\ &= \varepsilon\, r_i^T l_{i-1}^p + \varepsilon\, \gamma_i p_i^T \widetilde{\mathbf{A}}_i p_i - \varepsilon\, \widetilde{\zeta}_i^\gamma p_i^T \mathbf{A} p_i - \varepsilon\, \gamma_i p_i^T \widetilde{\mathbf{A}}_i p_i + \varepsilon\, p_i^T \widetilde{l}_i^r - \varepsilon^2 \widetilde{\zeta}_i^\gamma p_i^T \widetilde{\mathbf{A}}_i p_i \\ &= \varepsilon\, r_i^T l_{i-1}^p - \varepsilon\, \widetilde{\zeta}_i^\gamma p_i^T \mathbf{A} p_i + \varepsilon\, p_i^T \widetilde{l}_i^r - \varepsilon^2 \widetilde{\zeta}_i^\gamma p_i^T \widetilde{\mathbf{A}}_i p_i. \qquad (4.44) \end{aligned}$$

Z (4.44) a z (4.36) potom plyne

$$\varepsilon\, \frac{|M_i|}{\|r_i\|^2} \le \varepsilon\, \frac{\|l_{i-1}^p\|}{\|r_i\|} + \varepsilon\, |\widetilde{\zeta}_i^\gamma| \frac{p_i^T \mathbf{A} p_i}{\|r_i\|^2} + \varepsilon\, \frac{\|p_i\| \|\widetilde{l}_i^r\|}{\|r_i\|^2} + \varepsilon^2 \Big| \frac{p_i^T \widetilde{\mathbf{A}}_i p_i}{p_i^T \mathbf{A} p_i} \Big| \kappa(\mathbf{A})^{1/2}\, \mathcal{O}(n) + \mathcal{O}(\varepsilon^3).$$

Jelikož jsou velikosti výrazů vyskytující se u vyšších mocnin ε omezená, lze je dle našich konvencí zahrnout do $\mathcal{O}(\varepsilon^2)$, dostáváme

$$\varepsilon\, \frac{|M_i|}{\|r_i\|^2} \le \varepsilon\, \frac{\|l_{i-1}^p\|}{\|r_i\|} + \varepsilon\, |\widetilde{\zeta}_i^\gamma| \frac{p_i^T \mathbf{A} p_i}{\|r_i\|^2} + \varepsilon\, \frac{\|p_i\| \|\widetilde{l}_i^r\|}{\|r_i\|^2} + \mathcal{O}(\varepsilon^2). \qquad (4.45)$$

Odhadněme postupně výrazy na pravé straně (4.45). Z (4.28) plyne

$$\begin{aligned} \varepsilon\, \frac{\|l_{i-1}^p\|}{\|r_i\|} &\le \varepsilon\, \frac{\|r_i\| + 2\|\delta_{i-1} p_{i-1}\|}{\|r_i\|} + \mathcal{O}(\varepsilon^2) \\ &\le \varepsilon \left(1 + 2 \frac{\|p_i - r_i\|}{\|r_i\|} \right) + \mathcal{O}(\varepsilon^2) \\ &\le \varepsilon \left(3 + 2 \frac{\|p_i\|}{\|r_i\|} \right) + \mathcal{O}(\varepsilon^2). \qquad (4.46) \end{aligned}$$

Druhý výraz na pravé straně nerovnosti (4.45) odhadneme pomocí (4.36), platí

$$\varepsilon\, |\widetilde{\zeta}_i^\gamma| \frac{p_i^T \mathbf{A} p_i}{\|r_i\|^2} \le \varepsilon\, \kappa(\mathbf{A})^{1/2}\, \mathcal{O}(n) + \mathcal{O}(\varepsilon^2) \qquad (4.47)$$

a konečně z (4.35) plyne

$$\begin{aligned} \varepsilon\, \frac{\|p_i\| \|\widetilde{l}_i^r\|}{\|r_i\|^2} &\le \varepsilon \left(\frac{\|p_i\| \|r_i\| + 2\,\gamma_i \|p_i\| \|\mathbf{A} p_i\|}{\|r_i\|^2} \right) + \mathcal{O}(\varepsilon^2) \\ &= \varepsilon \left(\frac{\|p_i\|}{\|r_i\|} + 2 \frac{\|r_i\|^2}{p_i^T \mathbf{A} p_i} \frac{\|p_i\| \|\mathbf{A} p_i\|}{\|r_i\|^2} \right) + \mathcal{O}(\varepsilon^2) \\ &= \varepsilon \left(\frac{\|p_i\|}{\|r_i\|} + 2 \frac{\|p_i\| \|\mathbf{A} p_i\|}{\|p_i\|_{\mathbf{A}}^2} \right) + \mathcal{O}(\varepsilon^2) \end{aligned}$$

$$\leq \varepsilon \left(\frac{\|p_i\|}{\|r_i\|} + 2\,\kappa(\mathbf{A})^{1/2} \right) + \mathcal{O}(\varepsilon^2). \tag{4.48}$$

Velikost podílu $\|p_i\|/\|r_i\|$ lze odhadnout následovně:

$$\begin{aligned} \varepsilon \frac{\|p_i\|}{\|r_i\|} &\leq \varepsilon \left(\frac{\|r_i\| + \delta_{i-1}\|p_{i-1}\|}{\|r_i\|} \right) + \mathcal{O}(\varepsilon^2) \\ &\leq \varepsilon \left(1 + \frac{\|r_i\|^2}{\|r_{i-1}\|^2} \frac{\|p_{i-1}\|}{\|r_i\|} \right) + \mathcal{O}(\varepsilon^2) \\ &= \varepsilon \left(1 + \frac{\|r_i\|}{\|r_{i-1}\|} \frac{\|p_{i-1}\|}{\|r_{i-1}\|} \right) + \mathcal{O}(\varepsilon^2). \end{aligned}$$

Odhadneme-li stejným způsobem i podíly $\|p_{i-1}\|/\|r_{i-1}\|$, $\|p_{i-2}\|/\|r_{i-2}\|$, …, $\|p_1\|/\|r_1\|$ a užijeme-li faktu, že $\|p_0\|/\|r_0\| = 1$, dostáváme

$$\varepsilon \frac{\|p_i\|}{\|r_i\|} \leq \varepsilon \left(1 + \frac{\|r_i\|}{\|r_{i-1}\|} + \frac{\|r_i\|}{\|r_{i-2}\|} + \cdots + \frac{\|r_i\|}{\|r_0\|} \right) + \mathcal{O}(\varepsilon^2). \tag{4.49}$$

Z (4.25) a monotonie $\mathbf{A}$-normy chyby v konečné aritmetice plyne, že pro $k < i$ je

$$\varepsilon \frac{\|r_i\|}{\|r_k\|} \leq \varepsilon \frac{\lambda_{max}^{1/2}(\mathbf{A})}{\lambda_{min}^{1/2}(\mathbf{A})} \frac{\|x - x_i\|_{\mathbf{A}}(1 + \varepsilon F_i)}{\|x - x_k\|_{\mathbf{A}}(1 + \varepsilon F_k)} \leq \varepsilon\,\kappa(\mathbf{A})^{1/2} + \mathcal{O}(\varepsilon^2),$$

což dohromady s (4.49) dává

$$\varepsilon \frac{\|p_i\|}{\|r_i\|} \leq \varepsilon + i\,\varepsilon\,\kappa(\mathbf{A})^{1/2} + \mathcal{O}(\varepsilon^2). \tag{4.50}$$

Jednotlivé výrazy na pravé straně (4.45) jsme odhadli pomocí nerovností (4.46), (4.47), (4.48) a (4.50). Pro podíl $\varepsilon\,|M_i|/\|r_i\|^2$ platí

$$\varepsilon \frac{|M_i|}{\|r_i\|^2} \leq \varepsilon\,\kappa(\mathbf{A})^{1/2}\,\mathcal{O}(n+i) + \mathcal{O}(\varepsilon^2). \tag{4.51}$$

Ze vztahu (4.43) a nerovnosti (4.51) plyne

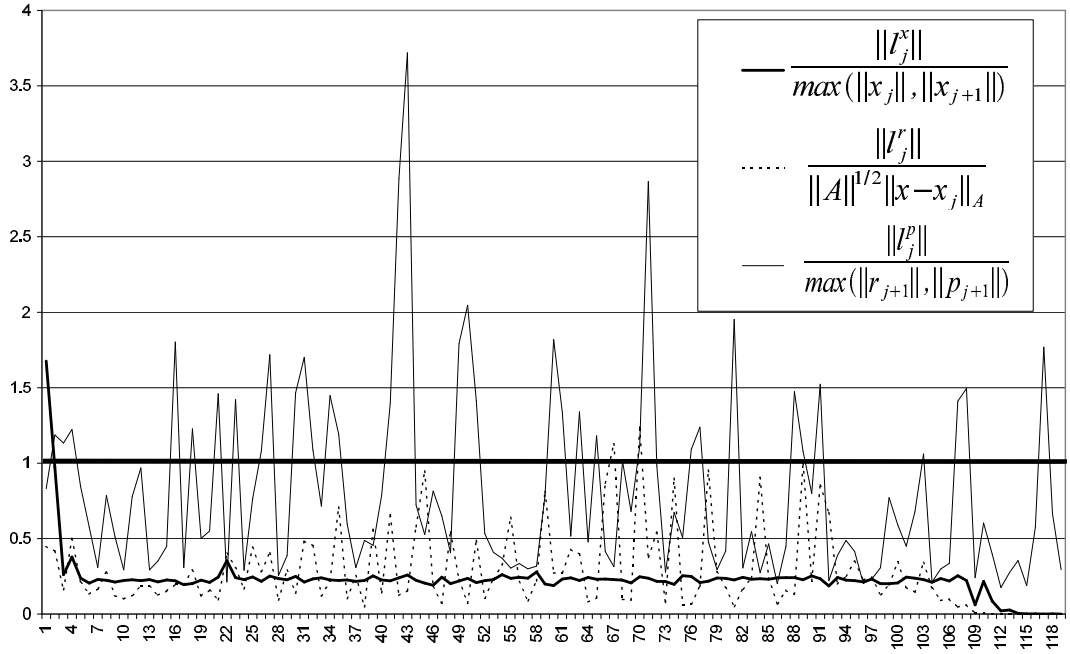
$$|p_j^T r_{j+1}| \leq \varepsilon\,\|r_j\|^2 \sum_{i=0}^{j} \frac{|M_i|}{\|r_i\|^2} + \mathcal{O}(\varepsilon^2) \leq \varepsilon\,\|r_j\|^2 \kappa(\mathbf{A})^{1/2}\,\mathcal{O}(j\,n + j^2/2) + \mathcal{O}(\varepsilon^2).$$

Věta je dokázána. □

4.5 Numerické experimenty

V numerických experimentech se věnujeme numerickému výpočtu lokálních zaokrouhlovacích chyb, které vznikají v (FP)CG a jejich odhadům. Ukážeme, že odhady (4.26)–(4.38) jsou při praktických výpočtech většinou velmi nadhodnoceny a na základě empirického pozorování se pokusíme určit realističtější odhady velikostí lokálních zaokrouhlovacích chyb.

Pro naše numerické experimenty opět použijeme soustavu ze strany 76, viz. (4.6). Připomeňme, že $n = 48$, $\lambda_{min} = 0.1$, $\lambda_{max} = 10^4$, $\kappa(\mathbf{A}) = 10^5$, $\kappa(\mathbf{A})^{1/2} \approxeq 316$. Popíšeme, jakým způsobem jsme vypočetli normy resp. velikosti lokálních chyb vznikajících při výpočtu vektorů resp. koeficientů v (FP)CG.



Obrázek 4.4: Velikosti lokálních chyb

Matici **A** vytvořenou v systému MATLAB jsme uložili do souboru, odkud jsme ji načetli do našeho programu v jazyce FORTRAN. V programu jsme počítali výsledky rekurencí CG v *dvojnásobné přesnosti* (double precision), strojová přesnost ε byla přibližně rovna

$$\varepsilon \approxeq 2.2 \times 10^{-16},$$

a dále v násobné aritmetice (balík MPFUN [5]) ve které jsme předepsali použití čísel s 32 platnými číslicemi, t.j. v aritmetice se strojovou přesností $\hat{\varepsilon} \sim \varepsilon^2$. Odečtením získaných výsledků jsme vypočetli lokální chyby (s nepřesností úměrnou $\mathcal{O}(\varepsilon^2)$) vznikající v původních rekurencích počítaných v dvojnásobné přesnosti. Právě uvedený popis výpočtu demonstrujme na konkrétním případě. Při počítání můžeme závislost mezi vektorem $x_j + \gamma_j p_j$ a výsledným vektorem x_{j+1}, který vznikne vypočtením v dvojnásobné přesnosti, vyjádřit vztahem

$$x_{j+1} = x_j + \gamma_j p_j + \varepsilon l_j^x.$$

Počítáme-li vektor $x_j + \gamma_j p_j$ v násobné aritmetice se strojovou přesností $\hat{\varepsilon}$, dostáváme vektor $\hat{x}_{j+1}$ a vztah mezi vektorem $x_j + \gamma_j p_j$ a $\hat{x}_{j+1}$ můžeme psát ve tvaru

$$\hat{x}_{j+1} = x_j + \gamma_j p_j + \hat{\varepsilon}\, \hat{l}_j^x.$$

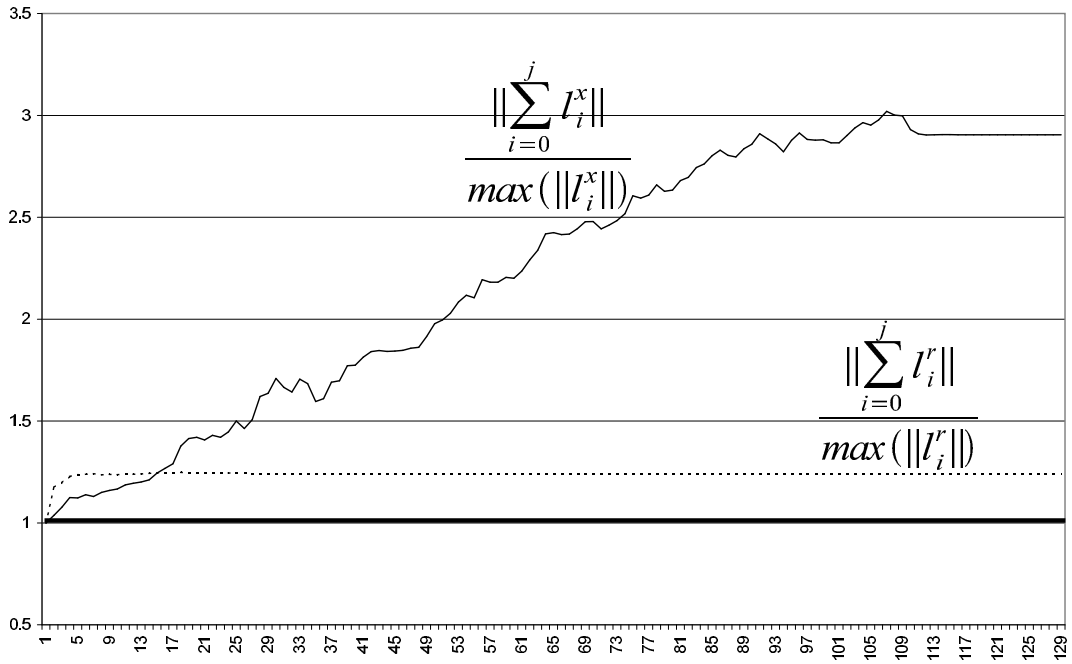
Protože je $\hat{\varepsilon} \sim \varepsilon^2$, dostaneme odečtením x_{j+1} a $\hat{x}_{j+1}$ vektor, který je až na nepřesnost velikosti $\mathcal{O}(\varepsilon^2)$ roven $\varepsilon\, l_j^x$. Vydělíme-li vektor $\varepsilon\, l_j^x$ strojovou přesností ε, obdržíme samotný vektor l_j^x. Podobně lze vypočíst všechny lokální chyby (vektory i skaláry) objevující se v (4.13)–(4.17).

Začneme odhadem lokálních chyb. Podle (4.26) je

$$\begin{aligned} \varepsilon \|l_j^x\| &\leq \varepsilon \{\|x_j\| + 2\|\gamma_j p_j\|\} + \mathcal{O}(\varepsilon^2) \\ &\leq \varepsilon \{\|x_j\| + 2\|x_{j+1} - x_j\|\} + \mathcal{O}(\varepsilon^2) \\ &\leq \varepsilon \max\{\|x_j\|, \|x_{j+1}\|\}\mathcal{O}(1) + \mathcal{O}(\varepsilon^2). \end{aligned}$$

Odhadneme-li podle (4.28) podobným postupem i hodnotu $\varepsilon \|l_j^p\|$, lze očekávat, že platí

$$\|l_j^x\| \leq \max\{\|x_j\|, \|x_{j+1}\|\}\mathcal{O}(1),$$



Obrázek 4.5: Jev krácení

$$\|l_j^p\| \quad \leq \quad \max\{\|r_{j+1}\|, \|p_{j+1}\|\}\mathcal{O}(1),$$

což nám potvrzuje i numerický experiment na obrázku 4.4, kde jsou hodnoty

$$\frac{\|l_j^x\|}{\max\{\|x_j\|, \|x_{j+1}\|\}} \qquad \text{a} \qquad \frac{\|l_j^p\|}{\max\{\|r_{j+1}\|, \|p_{j+1}\|\}}$$

znázorněny tučnou plnou křivkou a plnou křivkou normální tloušťky, které se po celou dobu iteračního procesu drží pod úrovní 4, t.j. jsou řádu $\mathcal{O}(1)$.

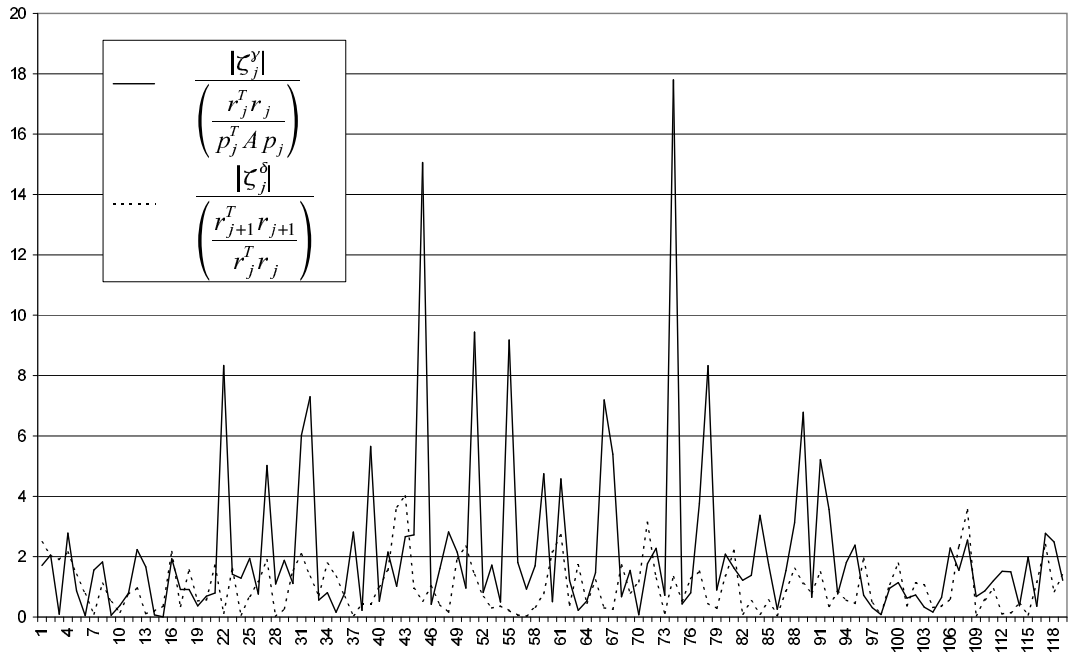
Když jsme v (4.29) odhadovali velikost normy chyby l_j^r pomocí $\mathbf{A}$-normy j-té chyby, způsobil největší problémy fakt, že obecně nejsme schopni efektivně odhadnout zaokrouhlovací chybu, která vzniká při násobení matice-vektor. Faktor $c\,\|\mathbf{A}\|\|\gamma_j p_j\|$ v (4.27) je většinou velmi nadhodnocen. Při odhadu tohoto výrazu pomocí $\mathbf{A}$-normy j-té chyby jsme byli navíc nuceni přejít od euklidovské normy vektoru k $\mathbf{A}$-normě vektoru, což s sebou přineslo další nadhodnocující faktory. Zatímco u prvního a druhého výrazu na pravé straně (4.27) jsme byli schopni ukázat úměrnost faktoru $\|\mathbf{A}\|^{1/2}\|x-x_j\|_{\mathbf{A}}$, třetí výraz jsme dokázali odhadnout pouze pomocí faktoru $\|\mathbf{A}\|^{1/2}\kappa(\mathbf{A})^{1/2}\|x-x_j\|_{\mathbf{A}}$. Numerické experimenty na obrázku 4.4 však ukazují, že velikost $\|l_j^r\|$ je při našich výpočtech úměrná hodnotě

$$\|\mathbf{A}\|^{1/2}\|x-x_j\|_{\mathbf{A}},$$

jak je vidět z čárkované křivky, kterou zakreslujeme podíl

$$\frac{\|l_j^r\|}{\|\mathbf{A}\|^{1/2}\|x-x_j\|_{\mathbf{A}}},$$

a faktor $\|\mathbf{A}\|^{1/2}\kappa(\mathbf{A})^{1/2}\|x-x_j\|_{\mathbf{A}}$ je zřejmě nadhodnocen. Čárkovaná křivka navíc zdaleka nedosahuje faktoru $\mathcal{O}(c)$ (v našem případě je $c = n^{3/2}$) ale pouze $\mathcal{O}(1)$. Zřejmě dochází k tomu, že sečtením dvou čísel reprezentujících zaokrouhlovací chyby o velikosti $\mathcal{O}(\varepsilon)$ je opět číslo o velikosti $\mathcal{O}(\varepsilon)$. Tento jev budeme nazývat termínem *krácení*. Pokusme se



Obrázek 4.6: Chyby při výpočtu koeficientů γ_j a δ_j

pojem krácení více formalizovat na příkladu sumy lokálních zaokrouhlovacích chyb

$$\varepsilon \sum_{i=0}^{j} l_i^x,$$

která se vyskytuje v (4.22) jako jedna z hlavních složek vektoru $\varepsilon\, f_{j+1}$ (rozdílu skutečného a rekurzivního rezidua). Složky vektorů v sumě obsažené jsou výsledkem nepřesného počítání v konečné aritmetice, jsou různě velké, kladné i záporné. Mezi vektory sumy jsou vektory s velkou normou, které jsou důležité a vektory s malou normou, jež nás nezajímají. Sčítají-li se dva vektory o přibližně stejné normě, vzniká často vektor, jehož norma se příliš neliší od norem sčítaných vektorů a potom lze očekávat, že při praktických výpočtech bude

$$\left\| \sum_{i=0}^{j} l_i^x \right\| \;\leq\; \max_{0\leq i\leq j} \|l_i^x\| \mathcal{O}(1) \tag{4.52}$$

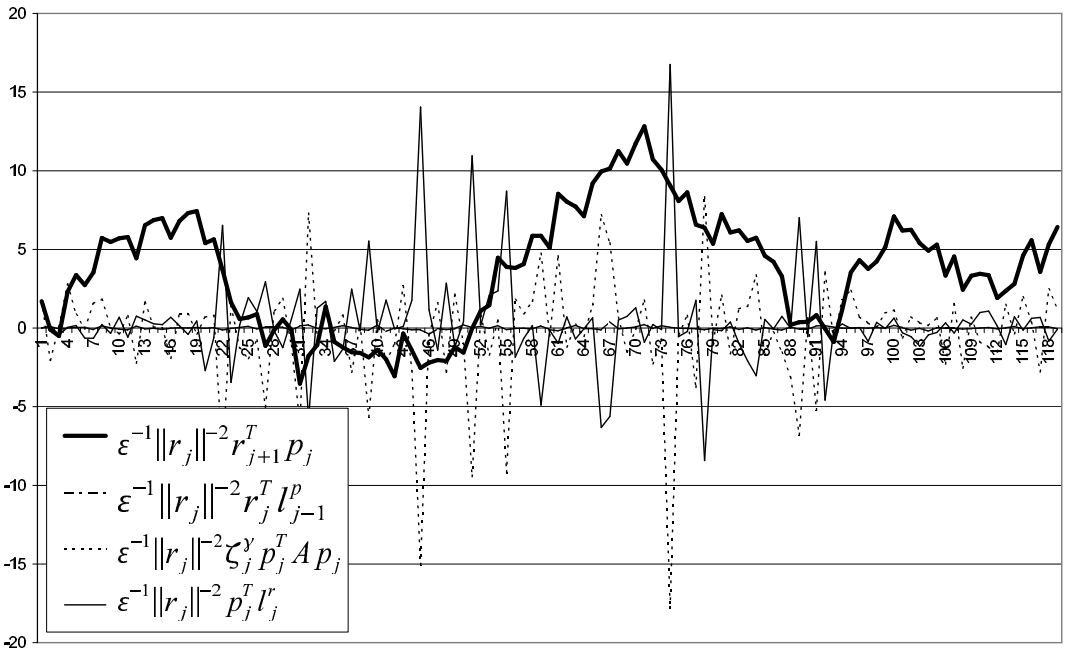
a analogicky pro sumu skalárů. Na obrázku 4.5 zakreslujeme plnou a čárkovanou křivkou hodnoty

$$\frac{\left\| \sum_{i=0}^{j} l_i^x \right\|}{\max\limits_{0\leq i\leq j} \|l_i^x\|} \qquad \text{a} \qquad \frac{\left\| \sum_{i=0}^{j} l_i^r \right\|}{\max\limits_{0\leq i\leq j} \|l_i^r\|}.$$

Je vidět, že ačkoliv podíly mohou velmi mírně růst, zůstává jejich velikost stále řádu $\mathcal{O}(1)$. Nyní je již jasné, že např. výraz v (4.38) typu $\mathcal{O}(j\,n + j^2/2)$ nebude hrát většinou významnou roli.

Vraťme se ale k odhadům velikosti lokálních chyb. Při výpočtu koeficientu δ_j se faktor $\mathcal{O}(n)$ uvedený v (4.31) díky krácení neprojevil, jak můžeme vidět na obrázku 4.6, kde čárkovanou křivkou zobrazujeme podíl

$$\frac{\delta_j}{\frac{\|r_{j+1}\|^2}{\|r_j\|^2}}.$$



Obrázek 4.7: Lokální ortogonalita

V (4.34) jsme zaokrouhlovací chybu vznikající při výpočtu koeficientu γ_j rozdělili na dvě části. Druhá část, kterou jsme si označili jako $\varepsilon\,\widetilde{\zeta}_j^\gamma$, je podle (4.36) úměrná $\varepsilon\,\gamma_j \kappa(\mathbf{A})^{1/2}$. V první části se vyskytuje problematický výraz

$$\frac{(p_j, \widetilde{\mathbf{A}}_j p_j)}{(p_j, \mathbf{A} p_j)},$$

který jsme nebyli schopni odhadnout lépe než pomocí faktoru $\kappa(\mathbf{A})$ a tento faktor se potom samozřejmě objevil ve výsledném odhadu velikosti chyby. V našem numerickém experimentu velikost čísla $|\zeta_j^\gamma|$ nepřevýšila hodnotu

$$\frac{\|r_j\|^2}{(p_j, \mathbf{A} p_j)} \kappa(\mathbf{A})^{1/2}, \tag{4.53}$$

jak je vidět z obrázku 4.6, na kterém znázorňujeme plnou křivkou podíl

$$\frac{|\zeta_j^\gamma|}{\frac{\|r_j\|^2}{(p_j, \mathbf{A} p_j)}}.$$

Zřejmě se opět projevilo krácení a faktor $\kappa(\mathbf{A})$, kterým odhadujeme velikost zlomku (4.53) je velmi nadhodnocen. Z obrázku 4.6 však zároveň plyne, že chyba vznikající při násobení matice-vektor má nezanedbatelný vliv na velikost zaokrouhlovací chyby vznikající při výpočtu koeficientu γ_j.

Dalším tématem, kterému se věnujeme v našich numerických experimentech je lokální ortogonalita. Na obrázku 4.7 zobrazujeme plnou tlustou křivkou podíl

$$\frac{r_{j+1}^T p_j}{\varepsilon\,\|r_j\|^2},$$

který jsme odhadli podle (4.38) hodnotou $\kappa(\mathbf{A})^{1/2}\,\mathcal{O}(n\,j)$. Jak se dalo očekávat, odhad (4.38) je nadhodnocen a absolutní hodnota podílu dosahuje maximální hodnoty 14. Dále

zobrazujeme jednotlivé složky čísla

$$\frac{M_j}{\|r_j\|^2} = \frac{r_j^T l_{j-1}^p}{\|r_j\|^2} - \frac{\zeta_j^\gamma p_j^T \mathbf{A} p_j}{\|r_j\|^2} + \frac{p_j^T l_j^r}{\|r_j\|^2}, \tag{4.54}$$

které hraje důležitou roli při odhadování velikosti skalárního součinu $r_{j+1}^T p_j$. Co je podstatné, vždy, když vzrostla velikost podílu

$$-\frac{\zeta_j^\gamma p_j^T \mathbf{A} p_j}{\|r_j\|^2},$$

zobrazeného čárkovanou křivkou, vzrostla zároveň velikost podílu

$$\frac{p_j^T l_j^r}{\|r_j\|^2},$$

vykresleného plnou křivkou normální tloušťky, v opačném směru a docházelo tedy k odečtení obou hodnot. To ovšem neznamená nic jiného, než že je podíl $M_j/\|r_j\|^2$ vyjádřen identitou (4.54) nevhodně. Proto jsme hledali způsob, jak rozložit poslední dva výrazy v identitě (4.54) tak, aby došlo k odečtení dominantních složek. Povšimli jsme si, že jak lokální chyba l_j^r tak i ζ_j^γ jsou zatíženy chybami, které vznikají při násobení matice $\mathbf{A}$ vektorem p_j. To nás vedlo k vyjádření lokálních chyb $\varepsilon\, l_j^r$ a $\varepsilon\, \zeta_j^\gamma$ ve tvarech (4.33) a (4.34), kde jsme v podstatě oddělili chybu způsobenou maticovým násobením od ostatních chyb. V (4.44) se poté díky tomuto vyjádření odečetl výraz $\varepsilon\, \gamma_i p_i^T \widetilde{\mathbf{A}}_i p_i$, který na obrázku 4.7 způsobil vznik velkých převýšení. Odečtení výrazu $\varepsilon\, \gamma_i p_i^T \widetilde{\mathbf{A}}_i p_i$ v (4.44) nám potom dopomohlo k těsnějšímu odhadu velikosti skalárního součinu $r_{j+1} p_j$. Pokud bychom odhadovali velikost hodnoty $M_j/\|r_j\|^2$ z původního tvaru (4.54), nevyhnuli bychom se faktoru $\kappa(\mathbf{A})$. Dodejme pro úplnost, že čerchovanou křivkou znázorňujeme podíl

$$\frac{r_j^T l_{j-1}^p}{\|r_j\|^2},$$

který zdá se nehraje významnou roli.

Z našich numerických experimentů plyne, že některé odhady velikosti lokálních zaokrouhlovacích chyb a rovněž odhad velikosti skalárního součinu $r_{j+1} p_j$ jsou nadhodnocené. Při počítání v konečné aritmetice se projevuje krácení, které způsobuje, že velikost faktorů typu $\mathcal{O}(n)$ je ve skutečnosti pouze $\mathcal{O}(1)$.

5

Odhady v konečné aritmetice

Odhady konvergenčních charakteristik, v našem případě **A***-normy chyby, mají za úkol získat z algoritmu informace, pomocí nichž zjistíme kvalitu aproximace řešení a na jejichž základě se rozhodujeme pro zastavení algoritmu. Je zřejmé, že zkoumání odhadů má dvě části, které od sebe nelze oddělovat. První částí je „teoretické" vymyšlení a ospravedlnění odhadu (viz. kapitola 3) a druhou analýza platnosti formule, dávající odhad, v konečné aritmetice. Bez druhé části bychom nevěděli, zda hodnoty, jež dostáváme z formulí, mají vůbec něco společného s aktuální* **A***-normou chyby. V této kapitole budeme diskutovat otázku použitelnosti odhadů v aritmetice s konečnou přesností. Vysvětlíme problém odhadů založených na pojetí* CG *jako Gaussovy kvadratury aplikovaných v aritmetice s konečnou přesností. U odhadů odvozených algebraickou cestou provedeme detailní analýzu zaokrouhlovacích chyb a použitelnost odhadů budeme demonstrovat v numerických experimentech. Součástí numerických experimentů bude i algebraické odvození odhadu euklidovské normy chyby a numerická demonstrace funkčnosti tohoto odhadu.*

Při aplikaci algoritmu CG v konečné aritmetice se globální ortogonalita mezi počítanými vektory po pár iteracích většinou úplně vytrácí a konvergenční charakteristiky (např. odhad **A**-normy chyby) založené na globální ortogonalitě či s její pomocí zformulované, mohou tudíž dávat informace absolutně nesouvisející se skutečnou konvergencí metody. Je zřejmé, že každou takovouto formuli je nutné analyzovat z hlediska zaokrouhlovacích chyb, jinak nevíme nic o korektnosti získávaných informací. Právě popsaný problém a nebezpečí dezinformace při použití nevhodných formulí pro počítání konvergenčních charakteristik přehlíží mnoho publikací o kterých budeme nyní hovořit. Zakládají své odhady na globální ortogonalitě mezi vektory, aniž by ospravedlňovaly použití nových formulí v aritmetice s konečnou přesností.

Práce Goluba a jeho spolupracovníků [8], o níž jsme hovořili v úvodu kapitoly 3 a která zahájila vývoj odhadů **A**-normy chyby v CG, byla důležitá také z jiného hlediska. Autoři poznamenávají, že pro zajištění stability počítaných uzlů a vah Gaussovy kvadratury je nutné bázi Krylovova prostoru reortogonalizovat. Byli si tedy dobře vědomi toho, že zaokrouhlovací chyby hrají důležitou roli v CG. V numerických experimentech uvedených v [8] však efekt zaokrouhlovacích chyb nebyl znatelný. Odhad normy chyby v iteračních metodách byl intenzivně studován v mnoha pracech, které následovali po [8] viz. např. [11], [17], [19], [18], [38], [39], [7]. Až na [19] však nebyl efekt zaokrouhlovacích chyb uvažován a publikovatelnost výsledku v praktických výpočtech zůstávala zcela neodůvodněna. Je pozoruhodné, že tento nedostatek poněkud zásadního rázu zůstal, až na [19], nepovšimnut.

V práci [15] autoři zdůraznili problém aplikace ideálních matematických formulí při počítání v aritmetice s konečnou přesností a navrhli užití intervalové aritmetiky pro počítání kontrolovaných odhadů. Toto pojetí však má, jak bylo poznamenáno v [15, p. 201], vážná a stěží překonatelná praktická omezení.

V práci [4] (zmínili jsme ji v úvodu kapitoly 3) autoři uvažovali předpodmíněnou metodu sdružených gradientů a uvedli (znovuobjevili) [32, (6:2)] nezávisle na [32]. Při

odvození použili vzájemné **A**-ortogonality směrových vektorů p_j, $j = 0, 1, \ldots, n-1$. Všimli si rovněž problémů spojených se ztrátou ortogonality a pozorovali, bez jakéhokoliv komentáře, že ačkoliv je jejich odhad odvozen na základě globální ortogonality, dává i při ztrátě ortogonality správné výsledky.

Problém aplikace ideálních matematických formulí při počítání v aritmetice s konečnou zůstával zřejmě nepovšimnut z toho důvodu, že ideální odhady odvozené v přesné aritmetice se zdají být potvrzeny experimenty v konečné aritmetice v tom smyslu, že odhadují aktuální **A**-normu chyby, která se může ve velikosti o několik řádů lišit od svého ideálního vzoru. Tato skutečnost byla na základě prací Paige, Greenbaumové a ostatních [44], [22], [28] vysvětlena v článku Goluba a Strakoše [19].

Naším cílem je rozšířit výsledky z [19], [59] a vysvětlit, že bez analýzy zaokrouhlovacích chyb není žádného ospravedlnění pro použití odhadů vyvinutých v přesné aritmetice při skutečných výpočtech.

Kapitola má následující strukturu. V sekci 5.1 se věnujeme odhadům založeným na Gaussově kvadratuře. Vysvětlíme, kdy a proč jsou odhady **A**-normy chyby vyvinuté v přesné aritmetice v dobré shodě s počítáním v konečné aritmetice. V sekci 5.2 je provedena detailní analýza zaokrouhlovacích v odhadech **A**-normy chyby odvozených algebraickou cestou. V 5.3 následují numerické experimenty, které se do detailů zabývají platností vztahu (3.29) v konečné aritmetice a demonstrují, které odhady jsou použitelné i v tom případě, kdy jsou lokální zaokrouhlovací chyby podstatně zesilovány v průběhu výpočtu.

5.1 Odhady založené na Gaussově kvadratuře

V této sekci se soustředíme na odhady **A**-normy chyby v CG, jež byly vyvinuty na základě Gaussovy kvadratury (nebo na její modifikaci) z integrálu (4.1), viz. např. algoritmus CGQL v [18]. Tento algoritmus je založen na vztahu (3.26), který předpokládá přesnou aritmetiku a zachování ortogonality. Při počítání v konečné aritmetice však dochází ke ztrátě ortogonality a nebylo zřejmé, zda má analogie (3.26) nějaký význam ve vztahu k hodnotám počítaným v konečné aritmetice. Vznikl fundamentální problém, který bylo nutné pochopit a řešit.

K pochopení tohoto problému výraznou měrou přispěla práce [19], založená na výsledcích z [22] a [28]. V [19] bylo dokázáno, že navzdory ztrátě ortogonality a lineární nezávislosti, základní vztah (3.26) (viz. rovněž (3.20)) platí rovněž pro *numericky počítané* hodnoty, s malou nepřesností úměrnou strojové přesnosti ε. Jinými slovy, za předpokladu, že další chyby v počítání odhadu založeného na Gaussově kvadratuře v CGQL z koeficientů CG jsou nepatrné, aproximuje tento algoritmus v j krocích v konečné aritmetice hodnotu (4.4). Tato hodnota se může podstatně lišit od hodnoty, která by byla vypočtena v přesné aritmetice ve stejném počtu kroků (4.3), avšak je ve shodě a aktuální normou chyby (FP)CG.

Práce [19] znamenala velký krok k pochopení fundamentálního problému odhadu **A**-normy chyby pomocí formulí založených na Gaussově kvadratuře v konečné aritmetice. K numerickému řešení problému odhadu **A**-normy chyby v konečné aritmetice však přispěla jen částečně. Z analýzy bylo zřejmé, že hodnotu označenou $\eta_{j,d}$ lze použít k odhadu kvadrátu **A**-normy j-té chyby s přesností $\mathcal{O}(\varepsilon)$. Po odmocnění celého vztahu však vyplynulo, že hodnotou $(\eta_{j,d})^{1/2}$ lze odhadovat **A**-normu j-té chyby pouze s přesností $\mathcal{O}(\varepsilon^{1/2})$, což potvrdili i numerické experimenty v [19] na soustavách, u kterých jsou lokální zaokrouhlovací chyby podstatně zesilovány v průběhu výpočtu algoritmu CG v konečné aritmetice, a tím byla značně omezena praktická použitelnost odhadu. V práci [18] byl navržen numericky vhodný postup výpočtu rozdílu $(\mathbf{T}_k^{-1})_{11} - (\mathbf{T}_j^{-1})_{11}$ který zajistil, že odhad založený na (3.26) je použitelný až po hladinu, kdy je aktuální **A**-norma chyby

úměrná strojové přesnosti ε a nikoli pouze $\varepsilon^{1/2}$ (algoritmus CGQL v [18]). Nebyla však provedena analýza zaokrouhlovacích chyb pro tento nový vztah.

Shrneme-li vše, v práci [19] byl vysvětlen fundamentální problém aplikace odhadů založených na Gaussově kvadratuře v konečné aritmetice a provedena analýza zaokrouhlovacích chyb u odhadu založeného na (3.26). Tato analýza prokázala možnost aplikace odhadu pouze po hladinu, kdy je $\mathbf{A}$-norma chyby úměrná $\varepsilon^{1/2}$. V [18] byl nalezen numericky vhodný postup, který umožnil odhadovat $\mathbf{A}$-normu chyby až po hranici úměrnou ε. Dosud však nebyla provedena analýza zaokrouhlovacích chyb, která by prokázala aplikovatelnost odhadů založených na (3.26) až po limitní hladinu přesnosti $\mathbf{A}$-normy chyby.

Jak obrázek 4.2 demonstruje, efekt zaokrouhlovacích chyb při počítání odhadů chyb vyvinutých na základě přesné aritmetiky může být velmi významný a tudíž nemůže být ignorován. Kdyby při počítání v konečné aritmetice nevznikaly žádné zaokrouhlovací chyby v Lanczosově a CG procesu, potom by nepřesnosti při počítání odhadů chyby (tak jako v CGQL) z CG nebo Lanczosových koeficientů, reprezentovali jediné nepřesnosti způsobené zaokrouhlovacími chybami a byly by vskutku nepatrné. Potom bychom v kroku j odhadovali bod na čerchované křivce z obrázku 4.2 a experimentální výsledky by byly v dobré shodě s *ideálními* z (3.20), (3.26) a (4.3). Při aplikaci odhadů v konečné aritmetice však musíme brát v úvahu zaokrouhlovací chyby vznikající během celého výpočtu. Je důležité si uvědomit, že odhady aplikované v konečné aritmetice zůstávají v dobré shodě s konvergencí CG v konečné aritmetice díky (4.4). Dodejme pouze, že v tomto případě odhadujeme bod na plné křivce. Experimenty publikované v několika článcích označených v úvodu této kapitoly neberou rozdíl analogický k rozdílu mezi plnou a čerchovanou křivkou na obrázku 4.2 v úvahu. V odstavcích 4.1 a 4.2 jsme vysvětlili, že pro daný iterační krok j mohou být body na plné a čerchované křivce na obrázku 4.2 (hodnoty (4.3) a (4.4)) velmi vzdálené. Rozdíl mezi (4.3) a (4.4) reprezentuje porovnání plné a čerchované křivky na obrázku 4.2 ve *vertikálním směru*. Povšimněme si, že pro $j > n$ je vztah (4.3) roven nule, zatímco (FP)CG může být stále v časném stádiu konvergence. Jak bylo popsáno v odstavci 4.2, vztah (4.5), viz. rovněž [45], plná křivka na obrázku 4.2 může být rovněž interpretována jako zpožděná čerchovaná křivka viz. obrázek 4.3.

5.2 Odhady založené na algebraické manipulaci

V této sekci provedeme detailní analýzu zaokrouhlovacích chyb ve vztazích (3.27)–(3.29), na nichž jsou založeny odhady odvozené algebraickou cestou.

5.2.1 Analýza odhadu A-normy chyby založeného na $\nu_{j,d}$

Analýza vztahu (3.34), která je obsahem tohoto oddílu, je poměrně hodně technicky náročná. V [59], kde je rovněž uvedena analýza tohoto vztahu, jsme kvůli maximálnímu zjednodušení a přehlednosti vynechali některé technické pasáže a z výsledku, který jsme v [59] obdrželi, je zřejmý hlavní důvod funkčnosti odhadu založeného na $\nu_{j,d}$. Zde předkládáme detailní analýzu zaokrouhlovacích chyb, ze které bude naprosto zřejmé, do jakého okamžiku lze vztahu (3.34) použít v konečné aritmetice ke konstrukci dolního odhadu $\mathbf{A}$-normy chyby. Vše podstatné shrnujeme ve větě 5.1. Důsledky této základní věty vysvětlujeme v následné diskusi, jež vyplňuje zbytek tohoto odstavce.

Věta 5.1 *Uvažujme standardní model aritmetiky s pohyblivou řádovou čárkou. Chyby vznikající při výpočtu algoritmu* CG *v konečné aritmetice nechť jsou charakterizovány vztahy* (4.13)–(4.17) *a nechť* $(c+n)\,\kappa(\mathbf{A})\,\varepsilon \ll 1$. *Potom pro přirozené číslo* d *platí*

$$\|x - x_j\|_{\mathbf{A}}^2(1+\varepsilon\,\Delta_{j,d}^{(1)}) - \|x - x_{j+d}\|_{\mathbf{A}}^2 = (1+\varepsilon\,\Delta_{j,d}^{(2)})\,\nu_{j,d} + \varepsilon\,2\,E_{j,d}^{\nu} + \mathcal{O}(\varepsilon^2), \tag{5.1}$$

kde $\nu_{j,d}$ je definováno v (3.34),

$$\nu_{j,d} = \sum_{i=j}^{j+d-1} \gamma_i \|r_i\|^2,$$

$\varepsilon\,\Delta^{(1)}_{j,d}$ a $\varepsilon\,\Delta^{(2)}_{j,d}$ jsou reálná čísla, jejichž velikost lze odhadnout vztahy

$$\varepsilon\,|\Delta^{(1)}_{j,d}| \leq \varepsilon\,\kappa(\mathbf{A})\,\mathcal{O}(c\,d) + \mathcal{O}(\varepsilon^2), \tag{5.2}$$

$$\varepsilon\,|\Delta^{(2)}_{j,d}| \leq \varepsilon\,\kappa(\mathbf{A})\,\mathcal{O}((j+d)\,n + (j+d)^2/2 + c) + \mathcal{O}(\varepsilon^2) \tag{5.3}$$

a

$$E^{\nu}_{j,d} \equiv (x - x_{j+d})^T f_{j+d} - (x - x_j)^T f_j. \tag{5.4}$$

Horní odhad velikosti chyby $\varepsilon\,E^{\nu}_{j,d}$ je úměrný strojové přesnosti ε a $\mathbf{A}$-normě j-té chyby,

$$|\varepsilon\,E^{\nu}_{j,d}| \leq \varepsilon\,\|x - x_j\|_{\mathbf{A}} \|\mathbf{A}\|^{1/2} \kappa(\mathbf{A})^{1/2} \Theta_{j+d}\,\mathcal{O}(c\,(j+d)) + \mathcal{O}(\varepsilon^2), \tag{5.5}$$

kde Θ_{j+d} je definováno v (4.20).

Důkaz. Zabývejme se nejdříve velikostí rozdílu $\|x - x_j\|^2_{\mathbf{A}} - \|x - x_{j+1}\|^2_{\mathbf{A}}$ v konečné aritmetice. Protože platí

$$\begin{aligned} \|x - x_j\|^2_{\mathbf{A}} &= \|x - x_{j+1} + x_{j+1} - x_j\|^2_{\mathbf{A}} \\ &= \|x - x_{j+1}\|^2_{\mathbf{A}} + \|x_{j+1} - x_j\|^2_{\mathbf{A}} + 2(x - x_{j+1})^T \mathbf{A}(x_{j+1} - x_j), \end{aligned}$$

můžeme rozdíl $\|x - x_j\|^2_{\mathbf{A}} - \|x - x_{j+1}\|^2_{\mathbf{A}}$ vyjádřit ve tvaru

$$\|x - x_j\|^2_{\mathbf{A}} - \|x - x_{j+1}\|^2_{\mathbf{A}} = \|x_{j+1} - x_j\|^2_{\mathbf{A}} + 2(x - x_{j+1})^T \mathbf{A}(x_{j+1} - x_j). \tag{5.6}$$

Upravujme postupně výrazy na pravé straně (5.6). Dosazením za x_{j+1} dostáváme

$$\begin{aligned} \|x_{j+1} - x_j\|^2_{\mathbf{A}} &= (\gamma_j p_j + \varepsilon\, l^x_j)^T \mathbf{A}(\gamma_j p_j + \varepsilon\, l^x_j) \\ &= \gamma_j^2 p_j^T \mathbf{A} p_j + 2\varepsilon\,\gamma_j p_j^T \mathbf{A} l^x_j + \varepsilon^2 \|l^x_j\|^2_{\mathbf{A}} \\ &= \gamma_j \left(\frac{\|r_j\|^2}{p_j^T \mathbf{A} p_j} + \varepsilon\,\zeta^{\gamma}_j \right) p_j^T \mathbf{A} p_j + 2\varepsilon\,\gamma_j p_j^T \mathbf{A} l^x_j + \mathcal{O}(\varepsilon^2) \\ &= \gamma_j \|r_j\|^2 + \varepsilon\,\gamma_j \zeta^{\gamma}_j p_j^T \mathbf{A} p_j + 2\varepsilon\,\gamma_j p_j^T \mathbf{A} l^x_j + \mathcal{O}(\varepsilon^2) \\ &= \gamma_j \|r_j\|^2 \left(1 + \varepsilon\,\zeta^{\gamma}_j \frac{p_j^T \mathbf{A} p_j}{\|r_j\|^2} \right) + 2\varepsilon\,\gamma_j p_j^T \mathbf{A} l^x_j + \mathcal{O}(\varepsilon^2) \end{aligned} \tag{5.7}$$

Druhý výraz z pravé strany (5.6) upravme následujícím způsobem

$$\begin{aligned} (x - x_{j+1})^T \mathbf{A}(x_{j+1} - x_j) &= (b - \mathbf{A}x_{j+1})^T (\gamma_j p_j + \varepsilon\, l^x_j) \\ &= (r_{j+1} - \varepsilon\, f_{j+1})^T (\gamma_j p_j) + \varepsilon\,(x - x_{j+1})^T \mathbf{A} l^x_j \\ &= \gamma_j r_{j+1}^T p_j - \varepsilon\,\gamma_j p_j^T f_{j+1} + \varepsilon\,(x - x_j - \gamma_j p_j)^T \mathbf{A} l^x_j + \mathcal{O}(\varepsilon^2) \\ &= \gamma_j r_{j+1}^T p_j - \varepsilon\,(x_{j+1} - x_j)^T f_{j+1} \\ &\quad + \varepsilon\,(x - x_j)^T \mathbf{A} l^x_j - \varepsilon\,\gamma_j p_j^T \mathbf{A} l^x_j + \mathcal{O}(\varepsilon^2) \\ &= \gamma_j r_{j+1}^T p_j - \varepsilon\,\gamma_j p_j^T \mathbf{A} l^x_j \\ &\quad + \varepsilon\left[(x - x_j)^T \mathbf{A} l^x_j - (x_{j+1} - x_j)^T f_{j+1} \right] + \mathcal{O}(\varepsilon^2). \end{aligned} \tag{5.8}$$

Sečteme-li podle (5.6) vztah (5.7) s dvojnásobkem rovnice (5.8), dostáváme po odečtení výrazu $2\varepsilon\,\gamma_j p_j^T \mathbf{A} l_j^x$ vyjádření

$$\begin{aligned} \|x-x_j\|_{\mathbf{A}}^2 - \|x-x_{j+1}\|_{\mathbf{A}}^2 &= \gamma_j\|r_j\|^2\left(1+\varepsilon\,\zeta_j^\gamma\frac{p_j^T\mathbf{A}p_j}{\|r_j\|^2}\right)+2\gamma_j r_{j+1}^T p_j \\ &\quad +2\varepsilon\left[(x-x_j)^T\mathbf{A}l_j^x-(x_{j+1}-x_j)^T f_{j+1}\right]+\mathcal{O}(\varepsilon^2). \end{aligned} \tag{5.9}$$

Upravme výraz na druhém řádku (5.9). Platí

$$\begin{aligned} & (x-x_j)^T\mathbf{A}l_j^x-(x_{j+1}-x_j)^T f_{j+1} \\ = \;& (x-x_j)\mathbf{A}l_j^x-(x-x_j)^T f_{j+1}+(x-x_{j+1})^T f_{j+1} \\ = \;& (x-x_j)\mathbf{A}l_j^x-(x-x_j)^T(f_j+l_j^r+\mathbf{A}l_j^x)+(x-x_{j+1})^T f_{j+1} \\ = \;& (x-x_{j+1})^T f_{j+1}-(x-x_j)^T f_j-(x-x_j)^T l_j^r. \end{aligned} \tag{5.10}$$

Dosadíme-li za výraz $(x-x_j)^T\mathbf{A}l_j^x-(x_{j+1}-x_j)^T f_{j+1}$ do (5.9), dostáváme

$$\begin{aligned} \|x-x_j\|_{\mathbf{A}}^2 - \|x-x_{j+1}\|_{\mathbf{A}}^2 &= \gamma_j\|r_j\|^2\left(1+\varepsilon\,\zeta_j^\gamma\frac{p_j^T\mathbf{A}p_j}{\|r_j\|^2}\right)+2\gamma_j r_{j+1}^T p_j \\ &\quad -2\varepsilon\,(x-x_j)^T l_j^r \\ &\quad +2\varepsilon\left[(x-x_{j+1})^T f_{j+1}-(x-x_j)^T f_j\right]+\mathcal{O}(\varepsilon^2). \end{aligned} \tag{5.11}$$

Rozepíšeme-li skalární součin $r_{j+1}^T p_j$ podle (4.42), dostáváme z (5.11)

$$\begin{aligned} \|x-x_j\|_{\mathbf{A}}^2 - \|x-x_{j+1}\|_{\mathbf{A}}^2 &= \gamma_j\|r_j\|^2\left(1+\varepsilon\left\{\zeta_j^\gamma\frac{p_j^T\mathbf{A}p_j}{\|r_j\|^2}+2\sum_{k=0}^{j}\frac{M_k}{\|r_k\|^2}\right\}\right) \\ &\quad -2\varepsilon\,(x-x_j)^T l_j^r \\ &\quad +2\varepsilon\left[(x-x_{j+1})^T f_{j+1}-(x-x_j)^T f_j\right]+\mathcal{O}(\varepsilon^2). \end{aligned} \tag{5.12}$$

Ekvivalentní vyjádření rozdílu $\|x-x_j\|_{\mathbf{A}}^2-\|x-x_{j+1}\|_{\mathbf{A}}^2$ lze nalézt i v našem článku [59, vztah (10.1)]. Vztah (5.12) lze odvodit z [59, (10.1)] rozepsáním některých vektorů podle příslušných rekurencí. Vyjádření (5.11) nám v následujícím umožní do detailů analyzovat vztah mezi rozdílem $\|x-x_j\|_{\mathbf{A}}^2-\|x-x_{j+d}\|_{\mathbf{A}}^2$ a hodnotou $\nu_{j,d}$ v konečné aritmetice.

Označme výraz ve složené závorce v (5.12) symbolem $\Delta_{j,1}^{(2)}$,

$$\Delta_{j,1}^{(2)} \equiv \zeta_j^\gamma\frac{p_j^T\mathbf{A}p_j}{\|r_j\|^2}+2\sum_{k=0}^{j}\frac{M_k}{\|r_k\|^2}.$$

Podle (4.51) a (4.32) zřejmě platí

$$\varepsilon\,\Delta_{j,1}^{(2)} \le \varepsilon\,\kappa(\mathbf{A})\,\mathcal{O}(j\,n+j^2/2+c). \tag{5.13}$$

Uvažujeme-li rovnice (5.12) pro iterační čísla $j,\ldots,j+d-1$, dostáváme jejich sečtením

$$\begin{aligned} \|x-x_j\|_{\mathbf{A}}^2 - \|x-x_{j+d}\|_{\mathbf{A}}^2 &= \sum_{i=j}^{j+d-1}\gamma_i\|r_i\|^2(1+\varepsilon\,\Delta_{i,1}^{(2)}) \\ &\quad -2\sum_{i=j}^{j+d-1}\varepsilon\,(x-x_i)^T l_i^r+2\varepsilon\,E_{j,d}^{\nu}+\mathcal{O}(\varepsilon^2). \end{aligned} \tag{5.14}$$

Definujme čísla $\Delta^{(1)}_{j,d}$ a $\Delta^{(2)}_{j,d}$ vztahy

$$\Delta^{(1)}_{j,d} \equiv 2 \sum_{i=j}^{j+d-1} \frac{(x-x_i)^T l_i^r}{\|x-x_j\|^2_{\mathbf{A}}}, \qquad \Delta^{(2)}_{j,d} \equiv 2 \sum_{i=j}^{j+d-1} \frac{\gamma_i \|r_i\|^2 \Delta^{(2)}_{i,1}}{\nu_{j,d}}.$$

Potom zřejmě můžeme psát (5.14) ve tvaru

$$\|x-x_j\|^2_{\mathbf{A}}(1+\varepsilon\,\Delta^{(1)}_{j,d}) - \|x-x_{j+d}\|^2_{\mathbf{A}} \;=\; (1+\varepsilon\,\Delta^{(1)}_{j,d})\,\nu_{j,d} + 2\varepsilon\,E^{\nu}_{j,d} + \mathcal{O}(\varepsilon^2). \tag{5.15}$$

Protože podle (4.29) platí

$$\begin{aligned} \varepsilon\,|(x-x_i)^T l_i^r| &= \varepsilon\,|(x-x_i)^T \mathbf{A}^{1/2}\mathbf{A}^{-1/2} l_i^r| \\ &\le \varepsilon\,\|\mathbf{A}^{-1/2}\|\|x-x_i\|_{\mathbf{A}}\|l_i^r\| \\ &\le \varepsilon\,\kappa(\mathbf{A})\|x-x_i\|^2_{\mathbf{A}}\,\mathcal{O}(c) + \|\mathbf{A}^{-1/2}\|\|x-x_i\|_{\mathbf{A}}\mathcal{O}(\varepsilon^2), \end{aligned}$$

můžeme $\varepsilon\,|\Delta^{(1)}_{j,d}|$ odhadnout následujícím způsobem

$$\varepsilon\,|\Delta^{(1)}_{j,d}| \le 2 \sum_{i=j}^{j+d-1} \frac{\varepsilon\,\kappa(\mathbf{A})\|x-x_i\|^2_{\mathbf{A}}\,\mathcal{O}(c)}{\|x-x_j\|^2_{\mathbf{A}}} + \mathcal{O}(\varepsilon^2) \le \varepsilon\,\kappa(\mathbf{A})\,\mathcal{O}(c\,d) + \mathcal{O}(\varepsilon^2). \tag{5.16}$$

V (5.16) jsme se dopustili malé nepřesnosti při manipulaci s výrazem $\mathcal{O}(\varepsilon^2)$. Tato nepřesnost je však plně ospravedlnitelná, uvědomíme-li si, že chyba $\mathcal{O}(\varepsilon^2)$ vznikající při odhadování $\|l_j^r\|$ v (4.29) je úměrná velikosti $\mathbf{A}$-normy chyby $\|x-x_j\|_{\mathbf{A}}$.

Jelikož platí $\gamma_i\|r_i\|^2 > 0$, lze $\varepsilon\,\Delta^{(2)}_{j,d}$ odhadnout s využitím (5.13),

$$\varepsilon\,|\Delta^{(2)}_{j,d}| \le \varepsilon\,2 \max_{j\le i<j+d} |\Delta^{(2)}_{i,1}| \le \varepsilon\,\kappa(\mathbf{A})\,\mathcal{O}((j+d)\,n + (j+d)^2/2 + c) + \mathcal{O}(\varepsilon^2)$$

Tím jsme dokázali první část věty, platnost vztahu (5.1). Zbývá ukázat (5.5). Z definice $E^{\nu}_{j,d}$ a z monotonie $\mathbf{A}$-normy chyby dostáváme

$$\begin{aligned} \varepsilon\,|E^{\nu}_{j,d}| &= \varepsilon\,|(x-x_{j+d})\mathbf{A}^{1/2}\mathbf{A}^{-1/2} f_{j+d} - \varepsilon\,(x-x_j)^T\mathbf{A}^{1/2}\mathbf{A}^{-1/2} f_j| \\ &\le \varepsilon\,\|x-x_{j+d}\|_{\mathbf{A}}\|f_{j+d}\|_{\mathbf{A}^{-1}} + \varepsilon\,\|x-x_j\|_{\mathbf{A}}\|f_j\|_{\mathbf{A}^{-1}} \\ &\le \varepsilon\,\|x-x_j\|_{\mathbf{A}}(\|f_{j+d}\|_{\mathbf{A}^{-1}} + \|f_j\|_{\mathbf{A}^{-1}}) \\ &\le \varepsilon\,\|x-x_j\|_{\mathbf{A}}\|\mathbf{A}^{-1/2}\|\|\mathbf{A}\|\Theta_{j+d}\,\mathcal{O}(c\,(j+d)) + \mathcal{O}(\varepsilon^2), \end{aligned}$$

přičemž v poslední nerovnosti jsme použili (4.19). Věta je dokázána. □

Důsledky věty 5.1. Ze vztahu (5.1) plyne, že v konečné aritmetice platí

$$\|x-x_j\|^2_{\mathbf{A}} - \|x-x_{j+d}\|^2_{\mathbf{A}} - 2\varepsilon\,E^{\nu}_{j,d} \approxeq \nu_{j,d} + \mathcal{O}(\varepsilon^2). \tag{5.17}$$

Kvalita aproximativní rovnosti v (5.17) je dána velikostí výrazů $\varepsilon\,\Delta^{(1)}_{j,d}$ a $\varepsilon\,\Delta^{(2)}_{j,d}$. Je-li například

$$\varepsilon\,\Delta^{(1)}_{j,d} \sim 10^{-11}, \qquad \varepsilon\,\Delta^{(2)}_{j,d} \sim 10^{-13},$$

lze očekávat, že čísla na pravé a levé straně aproximativní rovnosti (5.17) se budou shodovat přibližně na 11 platných cifer. Odhady (5.2) a (5.3) jsou zřejmě velmi nadhodnoceny, což potvrdí i numerické experimenty. Pokud bychom uvažovali, že výrazy typu $\mathcal{O}((j+d)\,n + (j+d)^2/2 + c)$ resp. $\mathcal{O}(c\,d\,)$ nehrají významnou roli, určoval by podle (5.2) a (5.3) kvalitu aproximativní rovnosti (5.17) řád čísla $\varepsilon\,\kappa(\mathbf{A})$.

Pokusme se dále analyzovat vztah (5.17). Chybu $E^{\nu}_{j,d}$,

$$E^{\nu}_{j,d} = (x - x_{j+d})^T f_{j+d} - (x - x_j)^T f_j,$$

lze rozdělit na dvě části, první příslušnou k $\|x - x_j\|_{\mathbf{A}}$ a druhou, která se svojí velikostí vztahuje k $\|x - x_{j+d}\|_{\mathbf{A}}$. Za použití nerovnosti

$$|(x - x_j)^T f_j| = |(x - x_j)^T \mathbf{A}^{1/2}\mathbf{A}^{-1/2} f_j| \le \|x - x_j\|_{\mathbf{A}} \|f_j\|_{\mathbf{A}^{-1}}$$

můžeme vyjádřit (5.17) ve tvaru

$$\|x - x_j\|^2_{\mathbf{A}} \left(1 + \frac{\|f_j\|_{\mathbf{A}^{-1}}}{\|x - x_j\|_{\mathbf{A}}} \mathcal{O}(\varepsilon)\right) - \|x - x_{j+d}\|^2_{\mathbf{A}} \left(1 + \frac{\|f_{j+d}\|_{\mathbf{A}^{-1}}}{\|x - x_{j+d}\|_{\mathbf{A}}} \mathcal{O}(\varepsilon)\right) \approxeq \nu_{j,d} + \mathcal{O}(\varepsilon^2).$$

Uvědomme si, že platí

$$\varepsilon \|f_j\|_{\mathbf{A}^{-1}} = \|\mathbf{A}^{-1} r_j - (x - x_j)\|_{\mathbf{A}}. \tag{5.18}$$

Pokud eventuálně uvažujeme, že po dosažení úrovně limitní hladiny přesnosti skutečného rezidua konverguje euklidovská i $\mathbf{A}^{-1}$ norma rekurzivního rezidua k nule, představuje podle (5.18) stagnující hodnota

$$\varepsilon \|f_j\|_{\mathbf{A}^{-1}}$$

úroveň limitní hladiny přesnosti $\mathbf{A}$-normy chyby. Podíl

$$\varepsilon \frac{\|f_j\|_{\mathbf{A}^{-1}}}{\|x - x_j\|_{\mathbf{A}}} = \frac{\|\mathbf{A}^{-1} r_j - (x - x_j)\|_{\mathbf{A}}}{\|x - x_j\|_{\mathbf{A}}} \tag{5.19}$$

je podle předchozí úvahy o významu hodnoty $\varepsilon \|f_j\|_{\mathbf{A}^{-1}}$ malý, dokud $\mathbf{A}$-norma j-té chyby nedosáhne limitní hladiny přesnosti. Kvalita aproximativní rovnosti

$$\|x - x_j\|^2_{\mathbf{A}} - \|x - x_{j+d}\|^2_{\mathbf{A}} \approxeq \nu_{j,d} \tag{5.20}$$

je zřejmě dána kvalitou aproximativních rovností (5.17) a

$$1 + \frac{\varepsilon \|f_{j+d}\|_{\mathbf{A}^{-1}}}{\|x - x_{j+d}\|_{\mathbf{A}}} \approxeq 1. \tag{5.21}$$

Zatímco kvalita aproximativní rovnosti (5.17) zůstává na přibližně stejné hladině po celou dobu výpočtu a je dána velikostí výrazů $\varepsilon\,\Delta^{(1)}_{j,d}$ a $\varepsilon\,\Delta^{(1)}_{j,d}$, kvalita aproximativní rovnosti (5.21) se zhoršuje úměrně řádové vzdálenosti počítané $\mathbf{A}$-normy chyby a limitní hladiny přesnosti $\mathbf{A}$-normy chyby. Lze tedy očekávat, že aproximativní rovnost (5.20) platí až do úrovně limitní hladiny přesnosti $\mathbf{A}$-normy chyby a počet shodných cifer na pravé a levé straně (5.20) klesá úměrně vzdálenosti od limitní hladiny přesnosti $\mathbf{A}$-normy chyby.

Z předchozího plyne, že hodnotu $\nu^{1/2}_{j,d}$ lze použít k odhadu $\mathbf{A}$-normy j-té chyby ve smyslu popsaném v sekci 3.3, dokud $\mathbf{A}$-norma $(j+d)$-té chyby nedosáhne limitní hladiny přesnosti. Pokud platí

$$\|x - x_j\|^2_{\mathbf{A}} \gg \|x - x_{j+d}\|^2_{\mathbf{A}},$$

je $\nu^{1/2}_{j,d}$ dobrou aproximací $\mathbf{A}$-normy j-té chyby, i za situace, že $\mathbf{A}$-norma $(j+d)$-té chyby již leží v hladině limitní přesnosti, t.j. $\|x - x_{j+d}\|_{\mathbf{A}} \sim \varepsilon \|f_{j+d}\|_{\mathbf{A}^{-1}}$. Uveďme příklad v konktrétních hodnotách. Předpokládejme hladinu limitní přesnosti $\mathbf{A}$-normy chyby úměrnou 10^{-14}, $\|x - x_j\|_{\mathbf{A}} \sim 10^{-13}$ a $\|x - x_{j+d}\|_{\mathbf{A}} \sim 10^{-14}$. Potom platí

$$\|x - x_j\|^2_{\mathbf{A}}(1 + \mathcal{O}(10^{-1})) + \|x - x_j\|^2_{\mathbf{A}}\,\mathcal{O}(10^{-2}) \approxeq \nu_{j,d}.$$

a odmocnina z pravé strany (t.j. $\nu^{1/2}_{j,d}$) je dobrým odhadem $\mathbf{A}$-normy j-té chyby s relativní odchylkou $\mathcal{O}(10^{-1})$.

Poznamenejme, že $\mathbf{A}$-norma chyby dosáhne limitní hladiny přesnosti přibližně ve stejné iteraci jako norma skutečného rezidua. Tvrzení plyne ze vztahu mezi oběma vektory $\mathbf{A}(x - x_j) = \mathbf{A}^{1/2}(\mathbf{A}^{1/2}(x - x_j))$.

5.2.2 Analýza odhadu A-normy chyby založeného na $\vartheta_{j,d}$

V předchozím odstavci jsme ukázali, že k odhadu **A**-normy chyby lze v aritmetice s konečnou přesností použít jednoduše počitatelnou hodnotu $\nu_{j,d}$. Mohlo by se zdát, že již není potřeba analyzovat jiné matematicky ekvivalentní odhady, které jsou více početně náročné a přitom dávají (v nejlepším případě) stejné výsledky. Věříme, že tomu tak není. Důvodů uvedeme hned několik.

Kromě klasických matematických důvodů jako jsou úplnost předkládané teorie a možnost aplikace řešeného problému k jiným účelům, o kterých zatím nevíme (všechno se může jednou hodit) je jedním z podstatných důvodů demonstrace důležitosti analýzy libovolné formule, která počítá nějakou konvergenční charakteristiku (nebo její odhad). Na příkladu

$$\vartheta_{j,d} = r_0^T(x_{j+d} - x_j) - r_j^T(x_j - x_0) + r_{j+d}^T(x_{j+d} - x_0)$$

uvidíme, že ačkoliv platnost vztahu (3.28) není založena na globální ba ani na lokální ortogonalitě, ale pouze na vztahu mezi rekurzivním a skutečným reziduem a tudíž by se mohlo zdát že „musí“ platit až do doby kdy norma skutečného rezidua dosáhne limitní hladiny přesnosti, lze $\vartheta_{j,d}$ použít k odhadu **A**-normy chyby pouze omezeně, dokud se **A**-norma chyby nepřiblíží k hranici úměrné $\varepsilon^{1/2}$. Potom již je hodnota $(\vartheta_{j,d})^{1/2}$ pro odhad nepoužitelná. Příčina tkví v počítání rozdílu $x_{j+d} - x_j$. Definujeme-li

$$\vartheta^\star_{j,d} \equiv r_0^T \sum_{i=j}^{j+d-1} \gamma_i p_i - r_j^T(x_j - x_0) + r_{j+d}^T(x_{j+d} - x_0), \tag{5.22}$$

t.j. počítáme-li uvedený rozdíl vektorů jiným způsobem, dojde k odečtení určitých zaokrouhlovacích chyb a najednou začne vše fungovat až do iterace, ve které dosáhne **A**-norma chyby, resp. skutečné reziduum limitní hladiny přesnosti.

Důležitou roli v našich úvahách bude hrát rozdíl $f_{j+d} - f_j$, který lze podle (4.22) vyjádřit ve tvaru

$$f_{j+d} - f_j = \varrho_j^d + \mathbf{A}\xi_j^d, \tag{5.23}$$

kde

$$\varrho_j^d \equiv \sum_{i=j}^{j+d-1} l_i^r, \qquad \xi_j^d \equiv \sum_{i=j}^{j+d-1} l_i^x.$$

V následující větě ukážeme, že velikosti zaokrouhlovacích chyb, které vystupují ve vztahu mezi $\vartheta_{j,d}$ resp. $\vartheta^\star_{j,d}$ a rozdílem kvadrátů **A**-norem neklesají resp. klesají úměrně velikosti **A**-normy j-té chyby.

Věta 5.2 *Uvažujme standardní model aritmetiky s pohyblivou řádovou čárkou. Chyby vznikající při výpočtu algoritmu* CG *v konečné aritmetice nechť jsou charakterizovány vztahy* (4.13)–(4.17). *Potom pro hodnoty $\vartheta_{j,d}$ a $\vartheta^\star_{j,d}$ definované v* (3.33) *a* (5.22) *platí*

$$\|x - x_j\|_{\mathbf{A}}^2 - \|x - x_{j+d}\|_{\mathbf{A}}^2 = \vartheta_{j,d} + \varepsilon\, E^\vartheta_{j,d} - \varepsilon\,(x - x_0)^T\mathbf{A}\xi_j^d, \tag{5.24}$$

$$\|x - x_j\|_{\mathbf{A}}^2 - \|x - x_{j+d}\|_{\mathbf{A}}^2 = \vartheta^\star_{j,d} + \varepsilon\, E^\vartheta_{j,d} + \mathcal{O}(\varepsilon^2), \tag{5.25}$$

přičemž $E^\vartheta_{j,d}$ je možno vyjádřit ve tvaru

$$E^\vartheta_{j,d} = (x - x_{j+d})^T(f_{j+d} + f_0) - (x - x_j)^T(f_j + f_0) - (x - x_0)^T\varrho_j^d. \tag{5.26}$$

*Velikost chyby $\varepsilon\, E^\vartheta_{j,d}$ je úměrná strojové přesnosti ε a **A**-normě j-té chyby, platí*

$$|\varepsilon\, E^\vartheta_{j,d}| \le \varepsilon\, \|x - x_j\|_{\mathbf{A}} \|\mathbf{A}\|^{1/2}\kappa(\mathbf{A})^{1/2}\Theta_{j+d}\,\mathcal{O}(c\,(j+d)) + \mathcal{O}(\varepsilon^2), \tag{5.27}$$

kde Θ_{j+d} je definováno v (4.20). *Mezi $E^{\nu}_{j,d}$ a $E^{\vartheta}_{j,d}$ platí vztah*

$$E^{\vartheta}_{j,d} = E^{\nu}_{j,d} - f_0^T(x_{j+d} - x_j) - (x - x_0)^T \varrho^d_j. \tag{5.28}$$

Důkaz. Podle odvození (3.22) platí

$$\|x - x_0\|^2_{\mathbf{A}} = (x - x_j)^T \mathbf{A}(x_j - x_0) + (x - x_0)^T \mathbf{A}(x_j - x_0) + \|x - x_j\|^2_{\mathbf{A}}. \tag{5.29}$$

Uvažujeme-li (5.29) pro iterační čísla j a $j + d$, dostáváme po odečtení obou rovnic a jednoduché algebraické manipulaci

$$\begin{aligned} \|x - x_j\|^2_{\mathbf{A}} - \|x - x_{j+d}\|^2_{\mathbf{A}} &= (x - x_0)^T \mathbf{A}(x_{j+d} - x_j) \\ &\quad - (x - x_j)^T \mathbf{A}(x_j - x_0) \\ &\quad + (x - x_{j+d})^T \mathbf{A}(x_{j+d} - x_0). \end{aligned}$$

Dosadíme-li za vektory $\mathbf{A}(x - x_i) = b - \mathbf{A}x_i$, $i = 0, j, j + d$, podle (4.18) vektor $r_i - \varepsilon\, f_i$, dostáváme

$$\begin{aligned} \|x - x_j\|^2_{\mathbf{A}} - \|x - x_{j+d}\|^2_{\mathbf{A}} &= (r_0 - \varepsilon\, f_0)^T(x_{j+d} - x_j) \\ &\quad - (r_j - \varepsilon\, f_j)^T(x_j - x_0) \\ &\quad + (r_{j+d} - \varepsilon\, f_{j+d})^T(x_{j+d} - x_0) \\ &= r_0^T(x_{j+d} - x_j) - \varepsilon\, f_0^T(x_{j+d} - x_j) \\ &\quad - r_j^T(x_j - x_0) + \varepsilon\, f_j^T(x_j - x_0) \\ &\quad + r_{j+d}^T(x_{j+d} - x_0) - \varepsilon\, f_{j+d}^T(x_{j+d} - x_0). \end{aligned} \tag{5.30}$$

Srovnání (5.30) s definicí $\vartheta_{j,d}$ vede na vyjádření

$$\begin{aligned} \|x - x_j\|^2_{\mathbf{A}} - \|x - x_{j+d}\|^2_{\mathbf{A}} &= \vartheta_{j,d} - \varepsilon\, f_0^T(x_{j+d} - x_j) + \varepsilon\, f_j^T(x_j - x_0) \\ &\quad -\varepsilon\, f_{j+d}^T(x_{j+d} - x_0) \\ &= \vartheta_{j,d} - \varepsilon\, f_0^T(x_{j+d} - x_j) + \varepsilon\, f_j^T(x_j - x_0) \\ &\quad -\varepsilon\, f_{j+d}^T(x_{j+d} - x_j) - \varepsilon\, f_{j+d}^T(x_j - x_0) \\ &= \vartheta_{j,d} \\ &\quad -\varepsilon\,(f_0 + f_{j+d})^T(x_{j+d} - x_j) - \varepsilon\,(f_{j+d} - f_j)^T(x_j - x_0). \end{aligned} \tag{5.31}$$

Přičteme-li a odečteme vektor x v druhých složkách skalárních součinů v (5.31) dostáváme roznásobením

$$\begin{aligned} \|x - x_j\|^2_{\mathbf{A}} - \|x - x_{j+d}\|^2_{\mathbf{A}} &= \vartheta_{j,d} \\ &\quad -\varepsilon\,(f_0 + f_{j+d})^T(x_{j+d} - x) - \varepsilon\,(f_0 + f_{j+d})^T(x - x_j) \\ &\quad +\varepsilon\,(f_{j+d} - f_j)^T(x - x_j) - \varepsilon\,(f_{j+d} - f_j)^T(x - x_0). \end{aligned}$$

Sečtení skalárních součinů s druhou složkou $x - x_j$ vede na vyjádření

$$\begin{aligned} \|x - x_j\|^2_{\mathbf{A}} - \|x - x_{j+d}\|^2_{\mathbf{A}} &= \vartheta_{j,d} - \varepsilon\,(f_0 + f_{j+d})^T(x_{j+d} - x) - \varepsilon\,(f_0 + f_j)^T(x - x_j) \\ &\quad -\varepsilon\,(f_{j+d} - f_j)^T(x - x_0). \end{aligned}$$

Rozepíšeme-li rozdíl $f_{j+d} - f_j$ podle (5.23) a definujeme-li

$$E^{\vartheta}_{j,d} \equiv (x - x_{j+d})^T(f_{j+d} + f_0) - (x - x_j)^T(f_j + f_0) - (x - x_0)^T \varrho^d_j,$$

dostáváme

$$\begin{aligned} \|x - x_j\|^2_{\mathbf{A}} - \|x - x_{j+d}\|^2_{\mathbf{A}} &= \vartheta_{j,d} + \varepsilon\,(f_0 + f_{j+d})^T(x - x_{j+d}) - \varepsilon\,(f_0 + f_j)^T(x - x_j) \\ &\quad -\varepsilon\,(\varrho^d_j + \mathbf{A}\xi^d_j)^T(x - x_0) \end{aligned}$$

$$= \vartheta_{j,d} + \varepsilon\, E^{\vartheta}_{j,d} - \varepsilon\,(x - x_0)^T \mathbf{A} \xi^d_j.$$

Ukázali jsme, že platí (5.24). Věnujme se nyní vztahu (5.25). Podle (4.14) je

$$x_{j+d} - x_j = \sum_{i=j}^{j+d-1} \gamma_i p_i + \varepsilon\, \xi^d_j. \tag{5.32}$$

Nahradíme-li ve vyjádření $\vartheta_{j,d}$ rozdíl vektorů $x_{j+d} - x_j$ podle (5.32) a užijeme-li definice $\vartheta^\star_{j,d}$, dostáváme

$$\begin{aligned} \vartheta_{j,d} &= \vartheta^\star_{j,d} + \varepsilon\, r_0^T \xi^d_j \\ &= \vartheta^\star_{j,d} + \varepsilon (x - x_0)^T \mathbf{A} \xi^d_j + \mathcal{O}(\varepsilon^2). \end{aligned} \tag{5.33}$$

Dosadíme-li $\vartheta_{j,d}$ ve tvaru (5.33) do vztahu (5.24), odečte se zřejmě výraz $\varepsilon (x - x_0)^T \mathbf{A} \xi^d_j$ a dostáváme (5.25). Vztah (5.28) obdržíme následující úpravou $E^{\vartheta}_{j,d}$:

$$\begin{aligned} E^{\vartheta}_{j,d} &= (x - x_{j+d})^T (f_{j+d} + f_0) - (x - x_j)^T (f_j + f_0) - (x - x_0)^T \varrho^d_j \\ &= (x - x_{j+d})^T f_{j+d} + f_0^T (x - x_{j+d}) - f_0^T (x - x_j) - (x - x_j)^T f_j - (x - x_0)^T \varrho^d_j \\ &= (x - x_{j+d})^T f_{j+d} - (x - x_j)^T f_j - f_0^T (x_{j+d} - x_j) - (x - x_0)^T \varrho^d_j \\ &= E^{\nu}_{j,d} - f_0^T (x_{j+d} - x_j) - (x - x_0)^T \varrho^d_j. \end{aligned}$$

Zbývá odhadnout chybu $\varepsilon\, E^{\vartheta}_{j,d}$. Protože platí

$$\begin{aligned} \varepsilon\, |(x - x_0)^T \varrho^d_j| &\le \varepsilon\, d \max_{j \le i \le j+d-1} \|l^r_i\|\, \|x - x_0\| \\ &\le \varepsilon\, \|x - x_j\|_{\mathbf{A}} \|\mathbf{A}^{-1/2}\| \|\mathbf{A}\| \|x - x_0\|\, \mathcal{O}(c\, d) + \mathcal{O}(\varepsilon^2) \\ &\le \varepsilon\, \|x - x_j\|_{\mathbf{A}} \|\mathbf{A}^{-1/2}\| \|\mathbf{A}\| \Theta_j\, \mathcal{O}(c\, d) + \mathcal{O}(\varepsilon^2), \end{aligned}$$

dostáváme z definice $E^{\vartheta}_{j,d}$

$$\begin{aligned} |\varepsilon\, E^{\vartheta}_{j,d}| &\le \varepsilon\, \|x - x_j\|_{\mathbf{A}} \|\mathbf{A}^{-1/2}\| (\|f_{j+d}\| + \|f_j\|) + \varepsilon\, |(x - x_0)^T \varrho^d_j| \\ &\le \varepsilon\, \|x - x_j\|_{\mathbf{A}} \|\mathbf{A}^{-1/2}\| \|\mathbf{A}\| \Theta_{j+d}\, \mathcal{O}(c\,(j + d)) + \mathcal{O}(\varepsilon^2). \end{aligned}$$

Věta je dokázána. □

Důsledky věty 5.2. Podle tvrzení věty 5.2 platí rovnost

$$\|x - x_j\|^2_{\mathbf{A}} - \|x - x_{j+d}\|^2_{\mathbf{A}} = \vartheta^\star_{j,d} + \varepsilon\, E^{\vartheta}_{j,d} + \mathcal{O}(\varepsilon^2),$$

a chyba $\varepsilon\, E^{\vartheta}_{j,d}$ je podle (5.27) úměrná strojové přesnosti ε, $\mathbf{A}$-normě j-té chyby a dalším faktorům, které souvisejí s vlastnostmi matice a s velikostí norem počítaných aproximací řešení. Podle (5.25) a (5.27) existuje číslo $C_{j,d}$ takové, že

$$|C_{j,d}| \le \|\mathbf{A}\|^{1/2} \kappa(\mathbf{A})^{1/2} \Theta_{j+d}\, \mathcal{O}(c\,(j + d))$$

a platí

$$\varepsilon\, E^{\vartheta}_{j,d} = -\varepsilon\, \|x - x_j\|_{\mathbf{A}} C_{j,d}.$$

Rovnici (5.25) potom můžeme přepsat ve tvaru

$$\|x - x_j\|^2_{\mathbf{A}} \left(1 + \frac{\varepsilon\, C_{j,d}}{\|x - x_j\|_{\mathbf{A}}}\right) - \|x - x_{j+d}\|^2_{\mathbf{A}} = \vartheta^\star_{j,d} + \mathcal{O}(\varepsilon^2). \tag{5.34}$$

Velikost čísla $\varepsilon\, C_{j,d}$ souvisí s limitní hladinou přesnosti **A**-normy chyby a proto lze očekávat, že aproximativní rovnost

$$1 + \frac{\varepsilon\, C_{j,d}}{\|x - x_j\|_{\mathbf{A}}} \approxeq 1 \tag{5.35}$$

bude splněna do té doby, dokud se aktuální **A**-normy chyb nepřiblíží k limitní hladině přesnosti **A**-normy chyby. Lze tedy očekávat, že platí aproximativní rovnost

$$\|x - x_j\|_{\mathbf{A}}^2 - \|x - x_{j+d}\|_{\mathbf{A}}^2 \;\approxeq\; \vartheta_{j,d}^{\star} \tag{5.36}$$

a počet shodných cifer na pravé a levé straně (5.36) klesá úměrně vzdálenosti od limitní hladiny přesnosti **A**-normy chyby. Hodnoty $(\vartheta_{j,d}^{\star})^{1/2}$ lze použít k odhadu **A**-normy j-té chyby, dokud $\|x - x_j\|_{\mathbf{A}}$ nedosáhne hladiny úměrné hladině limitní přesnosti **A**-normy chyby.

Věnujme se nyní analýze vztahu (5.24). Rozdíl kvadrátů **A**-norem j a $(j+d)$-té chyby se od hodnoty $\vartheta_{j,d}$ liší v konečné aritmetice podle (5.24) o chybu

$$\varepsilon\, E_{j,d}^{\vartheta} - \varepsilon\, (x - x_0)^T \mathbf{A} \xi_j^d + \mathcal{O}(\varepsilon^2).$$

Chyba $E_{j,d}^{\vartheta}$ nezpůsobuje žádný problém, protože je její velikost úměrná aktuální **A**-normě chyby. Velikost výrazu $-\varepsilon\, (x - x_0)^T \mathbf{A} \xi_j^d$ zřejmě není úměrná **A**-normě j-té chyby a platí

$$|\varepsilon\, (x - x_0)^T \mathbf{A} \xi_j^d| \le \varepsilon\, d\, \|x - x_0\|_{\mathbf{A}} \max_{j \le i \le j+d} \|x_i\|_{\mathbf{A}}.$$

S přibývajícími iteracemi lze očekávat, že

$$|\varepsilon\, (x - x_0)^T \mathbf{A} \xi_j^d| \sim \varepsilon\, \|x - x_0\|_{\mathbf{A}} \|x\|_{\mathbf{A}}.$$

Jakmile tedy **A**-norma chyby dosáhne hladiny úměrné $\varepsilon^{1/2}$ (t.j. kvadrát **A**-normy chyby dosáhne hladiny úměrné ε), stává se chyba $\varepsilon\, (x - x_0)^T \mathbf{A} \xi_j^d$ ve vztahu (5.24) dominantní a hodnota $\vartheta_{j,d}$ přestane být použitelná pro odhad **A**-normy chyby. Pod hladinou, která je úměrná $\varepsilon^{1/2}$ začne platit vztah

$$|\vartheta_{j,d}| \sim |\varepsilon\, (x - x_0)^T \mathbf{A} \xi_j^d|.$$

5.2.3 O odhadu A-normy chyby založeném na $\mu_{j,d}$

Pro úplnost uvádíme úvahu o použitelnosti hodnoty $\mu_{j,d}$ ke konstrukci odhadu **A**-normy chyby. Ze vztahu

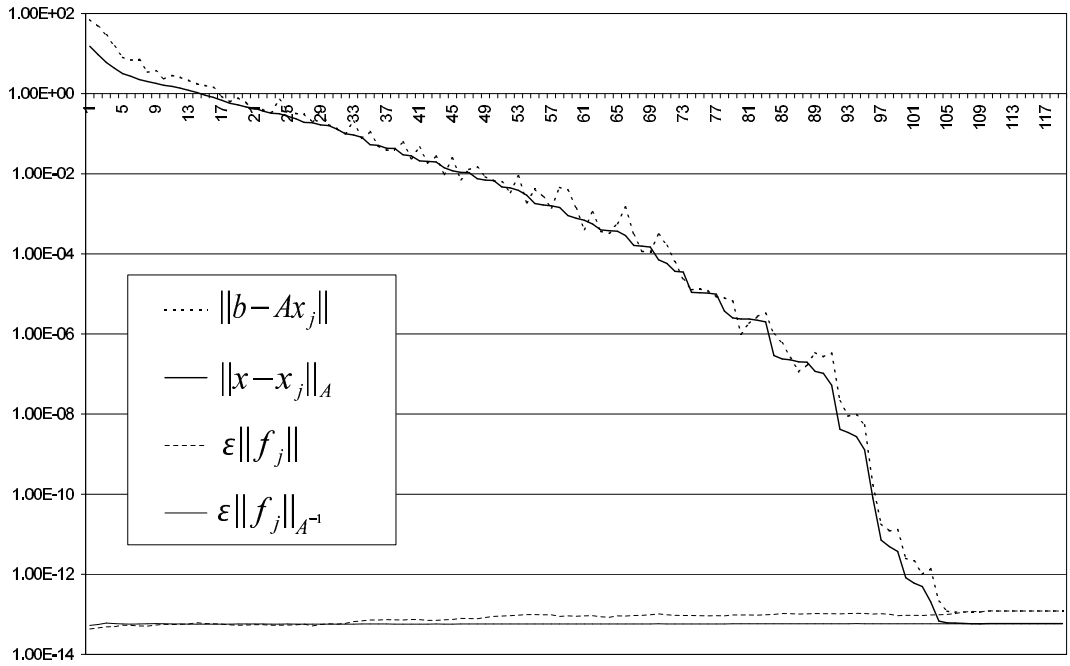
$$\vartheta_{j,d} = \mu_{j,d} - r_j^T (x_j - x_0) + r_{j+d}^T (x_{j+d} - x_0),$$

plyne na základě (5.24) rovnost

$$\|x - x_j\|_{\mathbf{A}}^2 - \|x - x_{j+d}\|_{\mathbf{A}}^2 = \mu_{j,d} - r_j^T (x_j - x_0) + r_{j+d}^T (x_{j+d} - x_0) + \varepsilon\, E_{j,d}^{\vartheta} - \varepsilon\, (x - x_0)^T \mathbf{A} \xi_j^d.$$

V předchozím odstavci jsme vysvětlili, že výraz $\varepsilon\, (x - x_0)^T \mathbf{A} \xi_j^d$ představuje potíže, na základě kterých mohl být odhad založený na $\vartheta_{j,d}$ použit pouze do úrovně, kdy **A**-norma j-té chyby klesla na hodnotu úměrnou $\varepsilon^{1/2}$. Tento výraz zmizí z pravé strany výše uvedené rovnice, počítáme-li rozdíl $x_{j+d} - x_j$ vhodným způsobem. Definujeme-li

$$\mu_{j,d}^{\star} \equiv r_0^T \sum_{i=j}^{j+d-1} \gamma_i p_i,$$



Obrázek 5.1: Limitní hladiny přesnosti

platí zřejmě

$$\|x - x_j\|_{\mathbf{A}}^2 - \|x - x_{j+d}\|_{\mathbf{A}}^2 = \mu_{j,d}^\star - r_j^T(x_j - x_0) + r_{j+d}^T(x_{j+d} - x_0) + \varepsilon\, E_{j,d}^\vartheta + \mathcal{O}(\varepsilon^2).$$

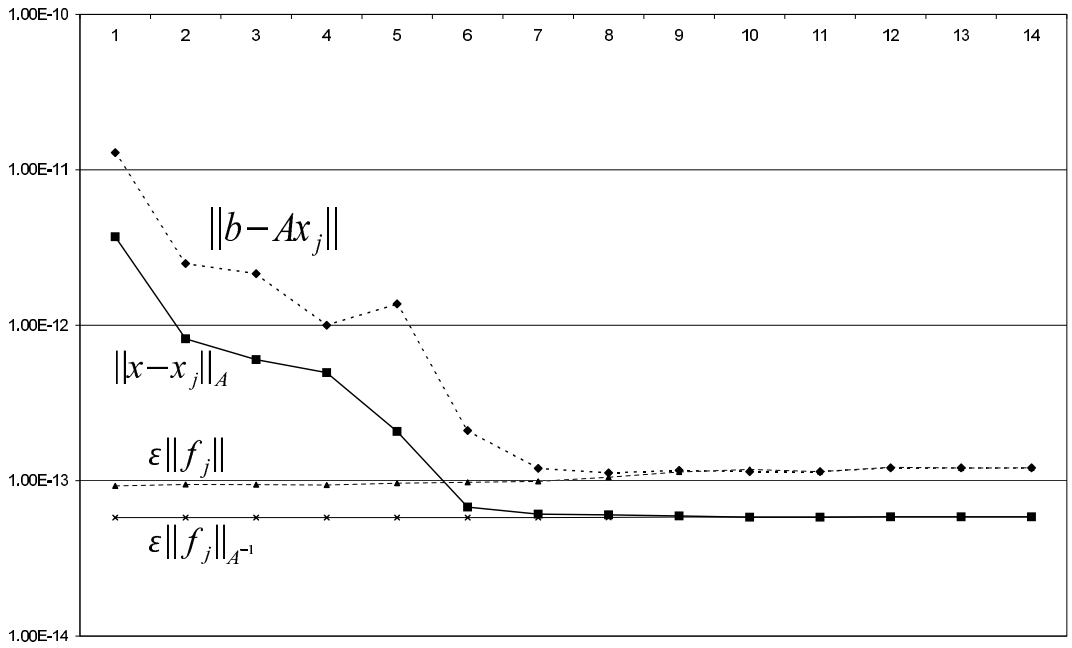
Výraz typu $r_j^T(x_j - x_0)$ je zřejmě svojí velikostí úměrný $\mathbf{A}$-normě j-té chyby (to zajišťuje přítomnost rezidua r_j). Úměrnost tohoto výrazu k strojové přesnosti ε však zřejmě závisí na tom, zda je reziduum r_j kolmé na všechny směrové vektory z předchozích iterací, tedy na kvalitě zachování globální ortogonality. Dojde-li při počítání v konečné aritmetice ke ztrátě globální ortogonality (a to nastává velmi často), nelze hodnot $\mu_{j,d}$ ani $\mu_{j,d}^\star$ použít k odhadu $\mathbf{A}$-normy chyby. Přesněji řečeno, $\mu_{j,d}^\star$ lze použít k odhadu $\mathbf{A}$-normy chyby přibližně do té doby, dokud platí $\|x - x_j\|_{\mathbf{A}} > \|\mathbf{I} - \mathbf{V}_j^T\mathbf{V}_j\|_F$.

5.3 Numerické experimenty

Numerické experimenty provádíme na stejném příkladu a se stejnou strojovou přesností, jak bylo uvedeno v sekci 4.5. Numericky prokážeme, že stagnující hodnota $\varepsilon\,\|f_j\|_{\mathbf{A}^{-1}}$ skutečně reprezentuje limitní hladinu přesnosti $\mathbf{A}$-normy chyby. Na našem konkrétním příkladě potvrdíme správnost úvah uvedených v důsledcích věty 5.1, budeme se zabývat aproximativní rovností (5.17). Ukážeme, že odhady založené na počítání $\nu_{j,d}$ a $\vartheta_{j,d}^\star$ jsou použitelné i v případě, kdy jsou lokální zaokrouhlovací chyby podstatně zesilovány v průběhu výpočtu algoritmu CG v konečné aritmetice. Na tomtéž příkladě budeme demonstrovat omezenost použití odhadů založených na $\mu_{j,d}$ a $\vartheta_{j,d}$.

V sekci 5.2 jsme velmi často hovořili o limitní hladině přesnosti skutečného rezidua a $\mathbf{A}$-normy chyby. Ukažme si nyní numericky, že limitní hladina přesnosti skutečného rezidua souvisí s velikostí hodnoty $\varepsilon\,\|f_j\|$ a hladina limitní přesnosti $\mathbf{A}$-normy chyby s hodnotou $\varepsilon\,\|f_j\|_{\mathbf{A}^{-1}}$, jak jsme naznačili v (5.19).

Na obrázku 5.1 je znázorněna norma skutečného reziduum $b - \mathbf{A}x_j$ (tučná čárkovaná křivka) a $\mathbf{A}$-norma chyby $\|x - x_j\|_{\mathbf{A}}$ (tučná plná křivka). Vidíme, že křivky $\varepsilon\,\|f_j\|$ (tenká čárkovaná) a $\varepsilon\,\|f_j\|_{\mathbf{A}^{-1}}$ (tenká plná) reprezentují limitní hladiny přesnosti skutečného



Obrázek 5.2: Detail - limitní hladiny přesnosti

rezidua a **A**-normy chyby. Poznamenejme, že vektor f_j a číslo $\|f_j\|_{\mathbf{A}^{-1}}$ jsme počítali v násobné aritmetice. Po celou dobu výpočtu jsou $\varepsilon\,\|f_j\|$ a $\varepsilon\,\|f_j\|_{\mathbf{A}^{-1}}$ velmi mírně rostoucí, drží se na téměř stejné hladině blízké strojové přesnosti ε. Jak jsme předpověděli v oddíle 5.2.1, norma skutečného rezidua a **A**-norma chyby dosáhnou svých limitních hladin přesnosti přibližně ve stejné iteraci, o čemž se můžeme přesvědčit na detailním pohledu obrázku 5.2.

Věnujme se nyní vztahu (5.1) a aproximativním rovnostem (5.17) a (5.20). V důsledcích věty 5.1 tvrdíme, že kvalita aproximativní rovnosti (5.17) (počet shodných číslic pravé a levé strany) je dána velikostí výrazů $\varepsilon\,\Delta^{(1)}_{j,d}$ a $\varepsilon\,\Delta^{(2)}_{j,d}$. Odhady (5.2) a (5.3) velikostí těchto hodnot jsou založeny na odhadech velikosti lokálních zaokrouhlovacích chyb $\varepsilon\,\zeta^{\gamma}_j$ a $\varepsilon\,l^r_j$, o kterých jsme v numerických experimentech předchozí kapitoly ukázali, že jsou velmi nadhodnoceny; při praktických výpočtech se místo faktoru $\kappa(\mathbf{A})$ resp. $\|\mathbf{A}\|$ projevuje pouze faktor $\kappa(\mathbf{A})^{1/2}$ resp. $\|\mathbf{A}\|^{1/2}$ a navíc dochází ke krácení. Proto lze očekávat, že velikosti výrazů $\varepsilon\,\Delta^{(1)}_{j,d}$ a $\varepsilon\,\Delta^{(2)}_{j,d}$ nepřevýší výrazně hodnotu $\varepsilon\,\kappa(\mathbf{A})^{1/2}$ a aproximativní rovnost (5.17) bude platit s vyšší přesností, než předpovídá věta 5.1 a vztahy (5.1)–(5.3).

Pro jednoduchost jsme uvažovali krok $d = 1$. V aritmetice se strojovou přesností $\hat{\varepsilon} \sim \varepsilon^3$ jsme vypočetli člen $\mathcal{O}(\varepsilon^2)$ z (5.15) a označili ho $\varepsilon^2\widetilde{E}^{\nu}_{j,d}$,

$$\varepsilon^2\widetilde{E}^{\nu}_{j,d} \equiv \|x-x_j\|^2_{\mathbf{A}}(1+\varepsilon\,\Delta^{(1)}_{j,d}) - \|x-x_{j+d}\|^2_{\mathbf{A}} - (1+\varepsilon\,\Delta^{(2)}_{j,d})\,\nu_{j,d} - 2\varepsilon\,E^{\nu}_{j,d} + \mathcal{O}(\varepsilon^3).$$

Graficky ukazujeme kvalitu tří aproximativních rovností

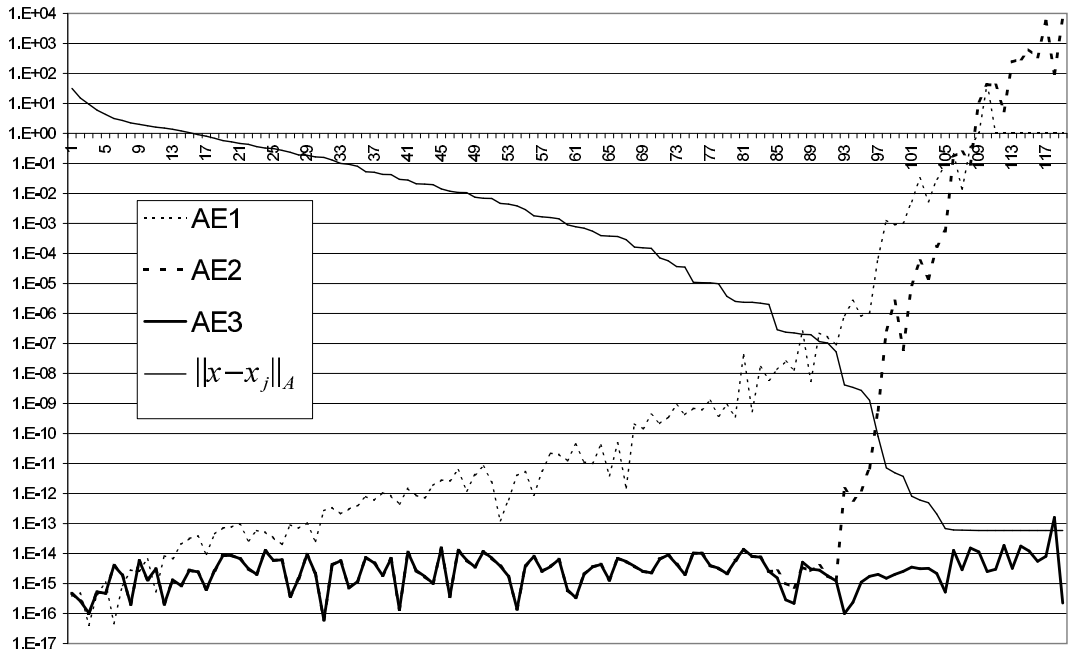
$$\|x-x_j\|^2_{\mathbf{A}} - \|x-x_{j+d}\|^2_{\mathbf{A}} \;\approxeq\; \nu_{j,d}, \tag{5.37}$$

$$\|x-x_j\|^2_{\mathbf{A}} - \|x-x_{j+d}\|^2_{\mathbf{A}} - 2\varepsilon\,E^{\nu}_{j,d} \;\approxeq\; \nu_{j,d}, \tag{5.38}$$

$$\|x-x_j\|^2_{\mathbf{A}} - \|x-x_{j+d}\|^2_{\mathbf{A}} - 2\varepsilon\,E^{\nu}_{j,d} \;\approxeq\; \nu_{j,d} + \varepsilon^2\widetilde{E}^{\nu}_{j,d}. \tag{5.39}$$

Na obrázku 5.3 znázorňujeme plnou křivkou normální tloušťky **A**-normu chyby $\|x-x_j\|_{\mathbf{A}}$. Normální čárkovanou křivkou zobrazujeme podíl

$$\frac{|\|x-x_j\|^2_{\mathbf{A}} - \|x-x_{j+d}\|^2_{\mathbf{A}} - \nu_{j,d}|}{\nu_{j,d}}, \tag{$AE1$}$$



Obrázek 5.3: Kvalita aproximativních rovností

který svým průběhem potvrzuje, že kvalita aproximativní rovnosti (5.37) se postupně zhoršuje s tím, jak se **A**-norma chyby přibližuje k limitní hladině přesnosti. Kvalita aproximativní rovnosti (5.38) vyjádřená podílem

$$(AE2) \qquad \frac{|\|x-x_j\|_{\mathbf{A}}^2 - \|x-x_{j+d}\|_{\mathbf{A}}^2 - \nu_{j,d} - 2\varepsilon\, E_{j,d}^{\nu}|}{\nu_{j,d}}$$

zakresleným tlustou čárkovanou křivkou se drží na hladině úměrné strojové přesnosti ε, dokud se nezačne v aproximativní rovnosti (5.38) projevovat člen $\mathcal{O}(\varepsilon^2)$.

Aproximativní rovnost (5.39) platí po celou dobu výpočtu. Pravá a levá strana (5.39) se shodují přibližně na počet cifer, který je dán řádem strojové přesnosti ε a tedy s vyšší přesností, než jsme předpověděli v důsledcích věty 5.1. To můžeme lehce vyčíst z obrázku 5.3 z tlusté plné křivky, kterou znázorňujeme podíl

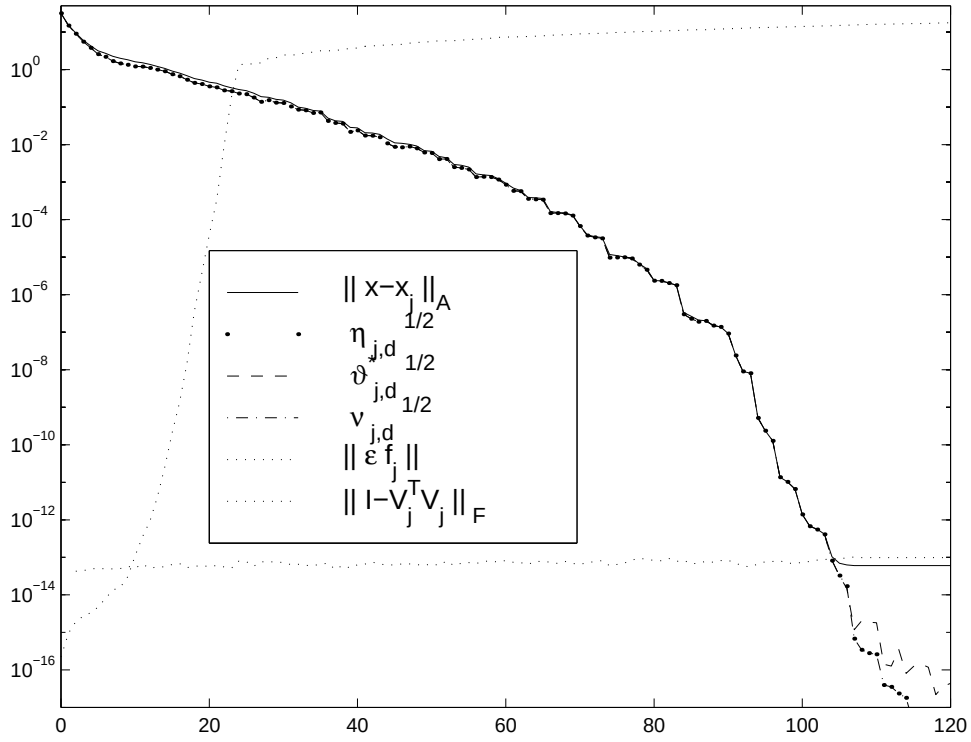
$$(AE3) \qquad \frac{|\|x-x_j\|_{\mathbf{A}}^2 - \|x-x_{j+d}\|_{\mathbf{A}}^2 - \nu_{j,d} - 2\varepsilon\, E_{j,d}^{\nu} - \varepsilon^2 \widetilde{E}_{j,d}^{\nu}|}{|\nu_{j,d} + \varepsilon^2 \widetilde{E}_{j,d}^{\nu}|}$$

vypovídající o kvalitě aproximativní rovnosti (5.39). Uvažujeme-li eventuálně, že po dosažení limitní hladiny přesnosti **A**-normy chyby konverguje rozdíl $\|x-x_j\|_{\mathbf{A}}^2 - \|x-x_{j+d}\|_{\mathbf{A}}^2$ a hodnota $\nu_{j,d}$ k nule, začne zřejmě platit aproximativní rovnost

$$-2\varepsilon\, E_{j,d}^{\nu} \approxeq \varepsilon^2 \widetilde{E}_{j,d}^{\nu}.$$

Věnujme se počítání odhadů **A**-normy chyby (programem MATLAB 5.1). Pro experimenty jsme zvolili konstantní krok $d = 4$.

Obrázek 5.4 ukazuje, že odhady ${\eta_{j,d}}^{1/2}$ (tečky), počítaný algoritmem CGQL [18], ${\vartheta_{j,d}^{\star}}^{1/2}$ (čárkovaná křivka), a ${\nu_{j,d}}^{1/2}$ (čerchovaná křivka) dávají, i za silného vlivu zaokrouhlovacích chyb na algoritmus, správné výsledky; všechny křivky jsou v podstatě totožné dokud $\|x-x_j\|_{\mathbf{A}}$ (plná křivka) nedosáhne své limitní hladiny přesnosti. Ztráta ortogonality, měřená pomocí $\|\mathbf{I}-\mathbf{V}_j^T\mathbf{V}_j\|_F$, je zakreslena příkře rostoucí tečkovanou křivkou. Vidíme,



Obrázek 5.4: Odhady použitelné v konečné aritmetice.

že pro $j \sim 22$ je ortogonalita mezi počítanými lanczosovými vektory úplně ztracena. Číslo $\|\varepsilon\, f_j\|$, které měří normu rozdílu skutečného a rekurzivního rezidua je znázorněno téměř vodorovnou tečkovanou křivkou, která zůstává blízká strojové přesnosti ε během celého výpočtu.

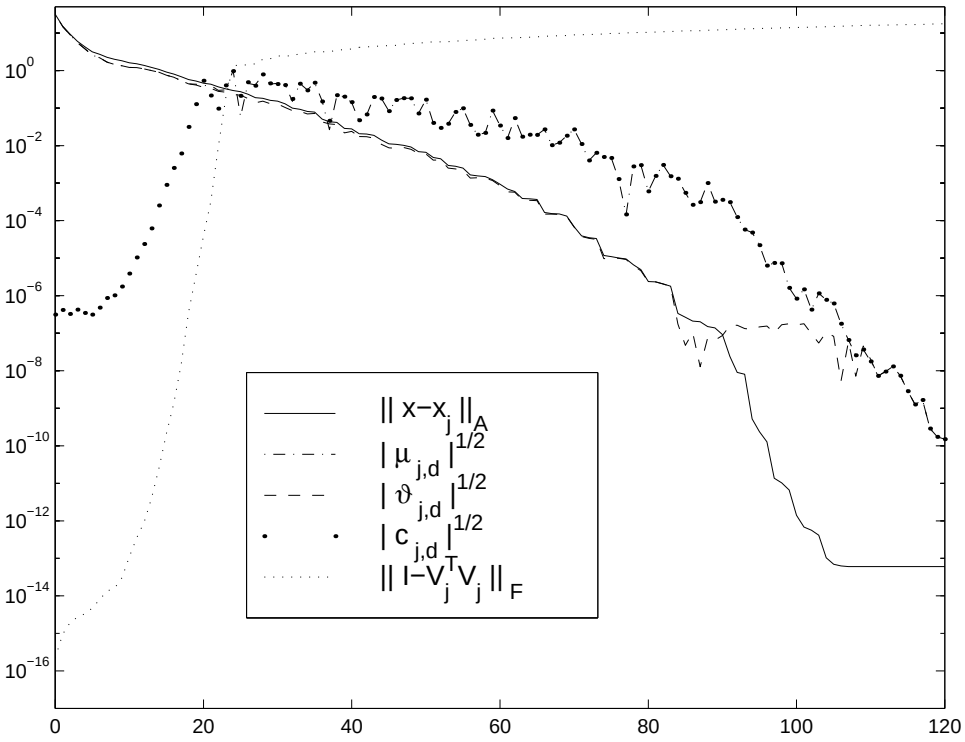
Na obrázku 5.5 vykreslujeme odhad počítaný pomocí $|\mu_{j,d}|^{1/2}$ (čerchovaná čára) a zdůrazňujeme důležitost korekčního výrazu

$$c_{j,d} \equiv -r_j^T(x_j - x_0) + r_{j+d}^T(x_{j+d} - x_0),$$

znázorněného tečkami. Jakmile je globální ortogonalita měřená $\|\mathbf{I} - \mathbf{V}_j^T\mathbf{V}_j\|_F$ a zakreslená tečkovanou křivkou ztracena, t.j. jakmile křivka ztráty ortogonality přetne křivku $\|x - x_j\|_{\mathbf{A}}$ (plná křivka), přestává odhad založený na $\mu_{j,d}$ fungovat, což je zřejmý důsledek závislosti vztahu (3.27) na globální ortogonalitě. Korekční výraz $c_{j,d}$ se stává dominantní, křivky $|c_{j,d}|^{1/2}$ a $|\mu_{j,d}|^{1/2}$ jsou od ztráty ortogonality v podstatě totožné a $|\mu_{j,d}|^{1/2}$ přestává vydávat rozumné výsledky korespondující s $\mathbf{A}$-normou chyby ($\mu_{j,d}$ dokonce nabývá záporných hodnot, proto počítáme odmocninu z absolutní hodnoty).

Pokud přičteme k $\mu_{j,d}$ korekční výraz $c_{j,d}$, získáme $\vartheta_{j,d}$ ($|\vartheta_{j,d}|^{1/2}$ je zobrazeno čárkovanou čárou). Jestliže však nepočítáme rozdíl $x_j - x_{j+d}$ způsobem, který byl popsán v analýze zaokrouhlovacích chyb, funguje odhad $|\vartheta_{j,d}|^{1/2}$ pouze do úrovně úměrné odmocnině ze strojové přesnosti (porovnejte $|\vartheta_{j,d}|^{1/2}$ na obrázku 5.5 s ${\vartheta^\star_{j,d}}^{1/2}$ na obrázku 5.4). Příčinou nefunkčnosti tedy není ztráta ortogonality, ale nevhodným způsobem počítaný rozdíl vektorů. Odmocninu opět počítáme z absolutní hodnoty čísla $\vartheta_{j,d}$, protože $\vartheta_{j,d}$ nabývá od určitých iterací záporných hodnot. To jen dokumentuje, že vydávané informace už nijak nekorespondují s $\mathbf{A}$-normou chyby.

Závěrem poznamenejme, že odhadu $\mathbf{A}$-normy chyby lze použít i k odhadu euklidovské



Obrázek 5.5: Odhady založené na $\mu_{j,d}$ a $\vartheta_{j,d}$.

normy chyby. Použijme výsledku z [32, Věta 6:3], platí

$$\|x - x_j\|^2 - \|x - x_{j+1}\|^2 = \frac{\|p_j\|^2}{(p_j, \mathbf{A}p_j)} \left(\|x - x_j\|_{\mathbf{A}}^2 + \|x - x_{j+1}\|_{\mathbf{A}}^2\right). \tag{5.40}$$

Sečtením rovnic (5.40) pro $j, \ldots, j + d - 1$ dostáváme

$$\|x - x_j\|^2 = \sum_{i=j}^{j+d-1} \frac{\|p_i\|^2}{(p_i, \mathbf{A}p_i)} \left(\|x - x_i\|_{\mathbf{A}}^2 + \|x - x_{i+1}\|_{\mathbf{A}}^2\right) + \|x - x_{j+d}\|^2. \tag{5.41}$$

Za předpokladu $\|x - x_j\|^2 \gg \|x - x_{j+d}\|^2$ je hodnota

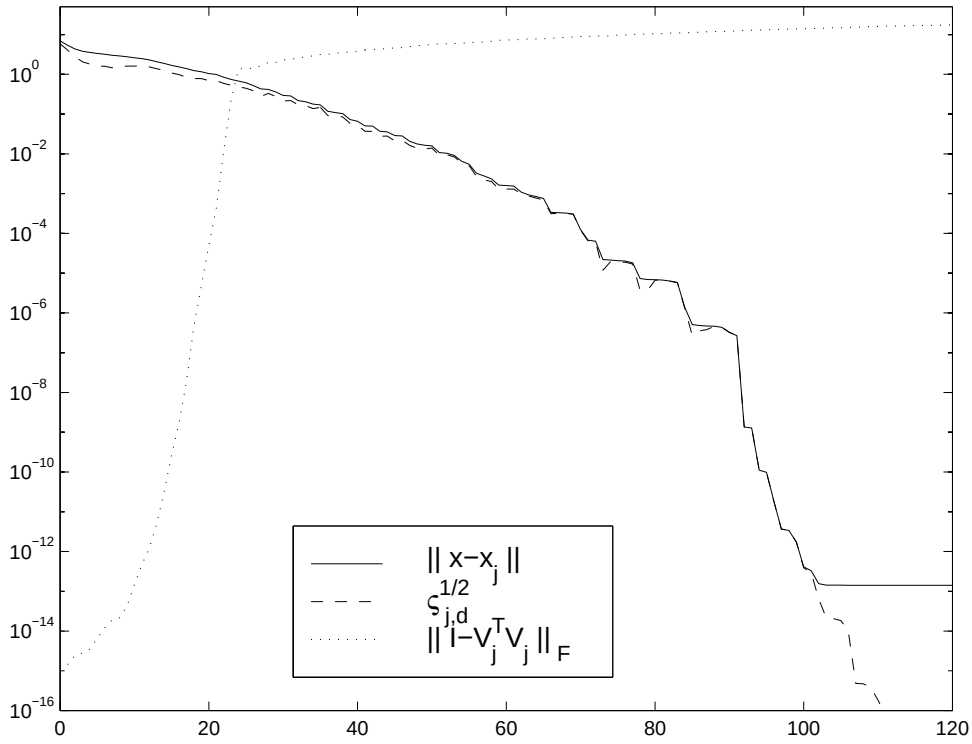
$$\sum_{i=j}^{j+d-1} \frac{\|p_i\|^2}{(p_i, \mathbf{A}p_i)} \left(\|x - x_i\|_{\mathbf{A}}^2 + \|x - x_{i+1}\|_{\mathbf{A}}^2\right) \tag{5.42}$$

dobrým dolním odhadem kvadrátu euklidovské normy chyby $\|x - x_j\|^2$. Hodnotu (5.42) neumíme sice vyjádřit přesně, ale můžeme ji zdola odhadnout tak, že nahradíme hodnoty $\|x - x_i\|_{\mathbf{A}}^2$ jejich dolními odhady. Uvědomme si, že chceme zároveň použít odhady pro kvadráty $\mathbf{A}$-norem chyb

$$\|x - x_j\|_{\mathbf{A}}^2, \ \ldots, \ \|x - x_{j+d}\|_{\mathbf{A}}^2.$$

Jestliže odhadneme $\|x - x_{j+d}\|_{\mathbf{A}}^2$ pomocí $\nu_{j+d,d}$, potřebujeme znát hodnoty $\gamma_i \|r_i\|^2$, $i = j + d, \ldots, j + 2d - 1$, kterých lze dále použít k přesnějšímu odhadu kvadrátů $\mathbf{A}$-norem chyb $\|x - x_i\|_{\mathbf{A}}^2$ pro $i < j + d$. Využijeme-li identit

$$\|x - x_j\|_{\mathbf{A}}^2 \quad = \quad \sum_{i=j}^{j+2d-1} \gamma_i \|r_i\|^2 + \|x - x_{j+2d}\|_{\mathbf{A}}^2,$$



Obrázek 5.6: Odhad euklidovské normy chyby

$$\begin{aligned}
\|x - x_{j+1}\|_{\mathbf{A}}^2 &= \sum_{i=j+1}^{j+2d-1} \gamma_i \|r_i\|^2 + \|x - x_{j+2d}\|_{\mathbf{A}}^2, \\
&\vdots \\
\|x - x_{j+d}\|_{\mathbf{A}}^2 &= \sum_{i=j+d}^{j+2d-1} \gamma_i \|r_i\|^2 + \|x - x_{j+2d}\|_{\mathbf{A}}^2,
\end{aligned}$$

a předpokladu $\|x - x_{j+d}\|_{\mathbf{A}}^2 \gg \|x - x_{j+2d}\|_{\mathbf{A}}^2$, je hodnota

$$\varsigma_{j,d} \equiv \sum_{i=j}^{j+d-1} \frac{\|p_i\|^2}{(p_i, \mathbf{A}p_i)} \left(\gamma_i \|r_i\|^2 + 2 \sum_{k=i+1}^{j+2d-1} \gamma_k \|r_k\|^2 \right) \tag{5.43}$$

dobrým dolním odhadem hodnoty (5.42) a tudíž i kvadrátu normy chyby $\|x - x_j\|^2$. Zřejmě tedy v iteračním kroku $j + 2d$ obdržíme odhad euklidovské normy chyby z kroku j (oproti odhadu $\mathbf{A}$-normy chyby potřebujeme dvojnásobný počet „předpočítaných“ kroků).

Na obrázku 5.6 znázorňujeme plnou křivkou euklidovskou normu chyby $\|x - x_j\|$ a čárkovanou křivkou odhad $\varsigma_{j,d}^{1/2}$ $(d = 4)$. Je zřejmé, že hodnota $\varsigma_{j,d}^{1/2}$ je dobrým odhadem euklidovské normy chyby $\|x - x_j\|$ kdykoliv dochází k dostatečně velkému poklesu normy i $\mathbf{A}$-normy chyby. Z numerického experimentu je vidět, že ztráta globální ortogonality (tečkovaná křivka) nemá vliv na aplikovatelnost odhadu. Pro ospravedlnění použití odhadu $\varsigma_{j,d}^{1/2}$ v konečné aritmetice bude nutné provést analýzu zaokrouhlovacích chyb ve vztahu (5.40), t.j. ukázat, že tento vztah platí i v konečné aritmetice s nějakou malou nepřesností, která je úměrná strojové přesnosti ε a velikosti aktuální normy chyby $\|x - x_j\|$.

Dosažené výsledky, závěry a shrnutí

Ve druhé kapitole jsme se zabývali vztahem metod s krátkými a dlouhými rekurencemi. Konkrétněji, uvažovali jsme Lanczosovu metodu pro řešení nesymetrických systémů lineárních rovnic (LM) a věnovali jsme se zkoumání parametru LM, stínovému vektoru, který ovlivňuje konvergenci této metody a hraje tedy zřejmě důležitou roli ve vztahu Lanczosovy metody a ostatních krylovovských metod.

Rozšířili jsme výsledky práce Greenbaum [25] a ve větě 2.3 jsme přesně rozlišili, kdy lze rekurence tříkrokové krylovovské metody považovat za rekurence Lanczosovy metody, t.j. kdy existuje stínový vektor takový, že LM počítá vektory dané tříkrokové rekurence. Pomocí stínového vektoru lze parametrizovat všechny třídiagonální matice podobné matici $\mathbf{A}$, které mají prvky ve vedlejších diagonálách nenulové (věta 2.4).

Ukázali jsme, že je možné určit stínový vektor tak, aby Lanczosova metoda vypočetla vybraná rezidua jiné krylovovské metody např. GMRES. Pokud však dlouhé rekurence uvažované krylovovské metody nemají speciální tvar, lze Lanczosovou metodou vypočíst nejvýše $\log_2(n)$ reziduí, což poukazuje na komplikovanost vztahu mezi Lanczosovou metodou a ostatními krylovovskými metodami.

V závěru druhé kapitoly jsme na příkladu teoretické krylovovské metody, která náhodně vybírá vektor z $k+1$ rozměrné jednotkové koule a určuje reziduum jako průnik vybraného směru s varietou reziduí, vysvětlili přirozenost faktu, že jsou konvergenční křivky klasických krylovovských metod často velmi blízké konvergenční křivce GMRES vždy, když dochází k jejímu dostatečně rychlému poklesu. Zároveň jsme vysvětlili, že náhodně volený stínový vektor je něco jiného než náhodně volené koeficiety v tříkrokové rekurenci.

V numerických experimentech druhé kapitoly jsme ukázali, že Lanczosova metoda může skutečně vypočíst vybraná rezidua metod FOM a GMRES. Vyjádřili jsme svůj názor, jakým způsobem je správné volit náhodný vektor. Numericky jsme určili stínový vektor optimální ve smyslu argumentu minima funkcionálu (2.40) a pozorovali jsme velmi blízké reziduové křivky QMR a GMRES.

Podle našeho názoru jsou, z teoretického hlediska, metody s krátkými rekurencemi velmi efektivní, protože s malými výpočetními náklady velmi často určují dobré aproximace řešení (blízké aproximacím metody GMRES*). Z praktického hlediska jsou fundamentálním problémem metod s krátkými rekurencemi zaokrouhlovací chyby, které mohou způsobit zpoždění konvergence ale i úplné znehodnocení výpočtu a je tedy velmi žádoucí pochopit chování těchto metod v konečné aritmetice počítače.*

V kapitolách 3, 4 a 5 a jsme se podrobně zabývali odhady $\mathbf{A}$-normy chyby v metodě sdružených gradientů a jejich použitelností v konečné aritmetice počítače. V kapitole 3 jsme ukázali, že odhady založené na Gaussově kvadratuře jsou matematicky ekvivalentní odhadům odvozeným algebraickou cestou a zformulovali jsme obecný princip odvozování odhadů $\mathbf{A}$-normy chyby (sekce 3.3) na základě znalosti hodnot z Gaussova kvadraturního vzorce (3.19).

Algebraickou cestou jsme odvodili nový odhad a upozornili jsme na fakt, že nejjednodušší odhad $\nu_{j,d}$ lze v podstatě nalézt již v původní práci [32]. Na příkladu odvození odhadu $\nu_{j,d}$ uvedeném v [4] jsme vysvětlili, že samotné použití globální ortogonality resp. $\mathbf{A}$-ortogonality při odvozování formule nemusí nutně vést k neplatnosti výsledného vztahu v konečné aritmetice. Nevhodná volba postupu při odvození formule však vždy komplikuje její analýzu v konečné aritmetice.

Matematický model CG v konečné aritmetice založený na pochopení CG ve smyslu Gaussovy kvadratury a princip zpoždění konvergence jsme prezentovali v kapitole 4.

Formálně jsme popsali zaokrouhlovací chyby vznikající při výpočtu algoritmu CG v konečné aritmetice a odhadli jsme jejich velikost. Ukázali jsme, že velikost skalárního součinu mezi j-tým směrovým vektorem a $(j+1)$-ním reziduem (lokální ortogonalitu) lze odhadnou pomocí hodnoty úměrné strojové přesnosti ε, kvadrátu normy j-tého rezidua a druhé odmocniny z čísla podmíněnosti matice $\mathbf{A}$ (věta 4.1).

V numerických experimentech této kapitoly jsme se zabývali jevem krácení a skutečnou velikostí zaokrouhlovacích chyb vznikajících při výpočtu v jednotlichých rekurencích. Vektory a skaláry reprezentující zaokrouhlovací chyby v jednotlivých rekurencích jsme počítali za použití násobné aritmetiky. Názorně jsme demonstrovali, že odhady lokálních zaokrouhlovacích chyb jsou při praktických výpočtech často velmi nadhodnocené a že lokální ortogonalita mezi směrovým vektorem a reziduem z následné iterace je splněna při praktických výpočtech mnohem lépe, než naznačuje výsledek naší věty 4.1.

V následující kapitole 5 jsme rozšířili analýzu zaokrouhlovacích chyb u odhadu $\nu_{j,d}$ uvedenou v našem článku [59] a provedli jsme rovněž analýzu zaokrouhlovacích chyb u nových odhadů $\vartheta^{\star}_{j,d}$, $\vartheta_{j,d}$.

Naše analýza prokázala, že odhady založené na $\nu_{j,d}$ a $\vartheta^{\star}_{j,d}$ lze použít i za situace, kdy se výpočty algoritmu CG v konečné aritmetice vlivem zaokrouhlovacích chyb velmi liší od svých ideálních vzorů. Odhady počítané z hodnot $\mu_{j,d}$ a $\vartheta_{j,d}$ jsou zatíženy chybami, které výrazným způsobem omezují jejich aplikovatelnost v konečné aritmetice.

Numerické experimenty kapitoly 5 potvrzují předpovědi, které jsme obdrželi při analýze zaokrouhlovacích chyb. Dále jsme numericky předvedli, že hodnota $\varepsilon\|f_j\|_{\mathbf{A}^{-1}}$ reprezentuje limitní hladinu přesnosti $\mathbf{A}$-normy chyby a ukázali jsme, že se kvalita aproximativní rovnosti

$$\|x - x_j\|^2_{\mathbf{A}} - \|x - x_{j+d}\|^2_{\mathbf{A}} \approxeq \nu_{j,d}$$

postupně zhoršuje s tím, jak se $\mathbf{A}$-norma chyby přibližuje k limitní hladině přesnosti. V závěru numerických experimentů jsme použili výsledků původní práce Hestenese a Stiefela [32] ke konstrukci odhadu euklidovské normy chyby a demonstrovali jsme funkčnost tohoto odhadu v prostředí konečné aritmetiky počítače.

Nejjednodušší cestou k odhadu $\mathbf{A}$-normy chyby je použít hodnoty $\nu_{j,d}$, jejíž vypočtení nevyžaduje v podstatě žádné operace navíc a je numericky stabilní až do dosažení limitní hladiny přesnosti $\mathbf{A}$-normy chyby. K uspokojivému řešení problému odhadu $\mathbf{A}$-normy chyby zbývá nalézt způsob, jak adaptivně volit délku kroku d tak, aby se zohlednila přesnost, s níž chceme $\mathbf{A}$-normu chyby odhadovat.

Poděkování

Je mnoho lidí, jimž bych rád poděkoval za jejich pomoc v uplynulých pěti letech a kteří buď přímo nebo nepřímo ovlivnili tuto práci.

V letech 1997–2000 jsem byl nejvíce ovlivněn svými spolupracovníky z katedry Numerické matematiky KNM MFF UK, kterým patří můj dík.

Zvláště pak děkuji Janu Zítkovi, který mi hodně pomohl v získání přehledu v oblasti krylovovských metod, snažil se mě podporovat v mé další práci a poskytl mi cenné rady a připomínky k textu této práce. Dále děkuji Karlu Najzarovi za přínosné konzultace v oblasti konečných prvků a trpělivost, se kterou mi poznatky předkládal a v neposlední řadě paní sekretářce na katedře Numeriky Evě Plandorové, vždy ochotné a nápomocné při jakkoliv pro mě těžkém administrativním úkonu.

Když jsem v roce 2000 nastoupil v Ústavu Informatiky ÚI AV ČR, ani jsem netušil, jak moc to ovlivní moji disertační práci.

Děkuji Miro Rozložníkovi a Mirkovi Tůmovi za jejich cenné rady, připomínky, odborné konzultace a podporu při práci, Hance Bílkové za ochotu, pomoc, dobrou náladu na pracovišti, ale i za odborné diskuse v oblastech zpracování textu a počítačové grafiky a Julovi Štullerovi, který mi umožnil pracovat v klidu a pohodě.

Můj největší dík potom patří Zdeňku Strakošovi. Velice hodnotná spolupráce s ním vedla k napsání třetí, čtvrté a páté kapitoly. Neustále mi poskytoval cenné rady, připomínky a sděloval mi své dlouholeté zkušenosti z oboru krylovovských metod. Děkuji mu za podrobné přečtení mých textů, za návrhy a připomínky k jejich obsahu a úpravě a za podporu, přízeň a čas, které mi věnoval.

A nakonec jsem si nechal to nejmilejší, co mě v životě potkalo, moji ženu Evu. Děkuju ji velmi za psychickou podporu a zázemí, které mi poskytovala při psaní této práce, za trpělivost se kterou snášela všechny mé nálady i za její laskavý humor a lásku.

Literatura

[1] M. Arioli, *Stopping criterion for the Conjugate Gradient algorithm in a Finite Element method framework*, submitted to Numerische Mathematik, (2001).

[2] M. Arioli and L. Baldini, *Backward error analysis of a null space algorithm in sparse quadratic programming*, SIAM J. Matrix Anal. Appl., 23 (2001), pp. 425–442.

[3] M. Arioli, V. Pták, and Z. Strakoš, *Krylov sequences of maximal length and convergence of GMRES*, BIT, 38 (1998), pp. 636–643.

[4] O. Axelsson and I. Kaporin, *Error norm estimation and stopping criteria in preconditioned Conjugate Gradient iterations*, Numerical Linear Algebra with Applications, 8 (2001), pp. 265–286.

[5] D. H. Bailey, *MPFUN: A multiple precision floating point computation package* (*Fotran–77*)*, http://www.netlib.org/mpfun/*, NASA Ames Research Center, USA, March 1995.

[6] C. Brezinski, M. Redivo-Zaglia, and H. Sadok, *Avoiding breakdown and near-breakdown in Lanczos type algorithms*, Numerical Algorithms, 1 (1991), pp. 261–284.

[7] D. Calvetti, S. Morigi, L. Reichel, and F. Sgallari, *Computable error bounds and estimates for the Conjugate Gradient method*, Numerical Algorithms, 25 (2000), pp. 79–88.

[8] G. Dahlquist, G. H. Golub, and S. G. Nash, *Bounds for the error in linear systems*, in Proc. Workshop on Semi-Infinite Programming, R. Hettich, ed., Springer, Berlin, 1978, pp. 154–172.

[9] O. G. Ernst, *Residual-minimizing Krylov subspace methods for stabilized discretization of convection-diffusion equations*, SIAM J. Matrix Anal. Appl., 21 (2000), pp. 1079–1101.

[10] V. Faber and T. Manteuffel, *Necessary and sufficient conditions for the existence of a conjugate gradient method*, SIAM J. Numer. Anal, 21 (1984), pp. 352–362.

[11] B. Fischer and G. Golub, *On the error computation for polynomial based iteration methods*, in Recent Advances in Iterative Methods, G. Golub, A. Greenbaum, and M. Luskin, eds., Springer, N.Y., 1994, pp. 59–67.

[12] R. W. Freund, *A transpose free quasi-minimal residual algorithm for non-Hermitian linear systems*, SIAM J.Sci.Statist. Comput., 13 (1993), pp. 470–492.

[13] R. W. Freund, M. H. Gutknecht, and N. M. Nachtigal, *An implementation of the look-ahead Lanczos algorithm for non-Hermitian matrices*, SIAM Journal on Scientific Computing, 14 (1993), pp. 137–158.

[14] R. W. Freund and N. M. Nachtigal, *QMR: a quasi-minimal residual method for non-hermitian linear systems*, Numer. Math., 60 (1991), pp. 315–339.

[15] A. Frommer and A. Weinberg, *Verified error bounds for linear systems through the Lanczos process*, Reliable Computing, 5 (1999), pp. 255–267.

[16] W. Gautschi, *A survey of Gauss-Christoffel quadrature formulae*, in E.B. Christoffel. The Influence of His Work on Mathematics and the Physical Sciences, P. Bultzer and F. Fehér, eds., Birkhauser, Boston, 1981, pp. 73–157.

[17] G. H. Golub and G. Meurant, *Matrices, moments and quadrature*, in Proceedings of the 15-th Dundee Conference, June 1993, D. Sciffeths and G. Watson, eds., Longman Sci. Tech. Publ., 1994, pp. 105–156.

[18] ——, *Matrices, moments and quadrature II: How to compute the norm of the error in iterative methods*, BIT, 37 (1997), pp. 687–705.

[19] G. H. Golub and Z. Strakoš, *Estimates in quadratic formulas*, Numerical Algorithms, 8 (1994), pp. 241–268.

[20] G. H. Golub and C. van Loan, *Matrix Computation*, The Johns Hopkins University Press, Baltimore MD, third ed., 1996.

[21] J. F. Grcar, *Analyses of the Lanczos Algorithm and of the Approximation Problem in Richardson's Method*, PhD thesis, Department of Computer Science, University of Illinois, Urbana, IL, 1981.

[22] A. Greenbaum, *Behavior of slightly perturbed Lanczos and Conjugate Gradient recurrences*, Lin. Alg. Appl., 113 (1989), pp. 7–63.

[23] ——, *Estimating the attainable accuracy of recursively computed Residual methods*, SIAM J. Matrix Anal. Appl., 18 (3) (1997), pp. 535–551.

[24] ——, *Iterative Methods for Solving Linear Systems*, SIAM, Philadelphia, 1997.

[25] ——, *On the role of the left starting vector in the two-sided Lanczos algorithm and nonsymetric linear system solvers*, in Proceedings of the Dundee meeting in Numerical Analysis, D. Griffiths, D. Highham, and G. Watson, eds., Pitman Research Notes in Mathematics Series 380, Longman, 1997.

[26] ——, *Relation of a two-term recurrence for nonhermitian linear systems to GMRES*, tech. report, University of Washington, July 1, 1999.

[27] A. Greenbaum, V. Pták, and Z. Strakoš, *Any nonincreasing convergence curve is possible for GMRES*, SIAM Journal on Matrix Analysis and Applications, 17 (1996), pp. 465–469.

[28] A. Greenbaum and Z. Strakoš, *Predicting the behavior of finite precision Lanczos and Conjugate Gradient computations*, SIAM J. Matrix Anal. Appl., 18 (1992), pp. 121–137.

[29] ——, *Matrices that generate the same Krylov varieties*, in Recent Advances in Iterative Methods, G. Golub et al., ed., IMA Volumes in Maths and Its Applications, Springer, 1994, pp. 95–119.

[30] M. H. Gutknecht, *Lanczos-type solvers for nonsymmetric linear system of equations*, Technical Report TR–97–04, Swiss Center for Scientific Computing ETH-Zentrum, Switzerland, 1997.

[31] M. H. Gutknecht and Z. Strakoš, *Accuracy of two three-term and three two-term recurrences for krylov space solvers*, SIAM J. Matrix Anal. Appl., 22 (2001), pp. 213–229.

[32] M. R. Hestenes and E. Stiefel, *Methods of conjugate gradients for solving linear systems*, J. Res. Nat. Bureau Standarts, 49 (1952), pp. 409–435.

[33] N. J. Higham, *Accuracy and stability of numerical algorithms*, SIAM, Philadelphia, PA, 1996.

[34] W. D. Joubert, *Lanczos methods for the solution of nonsymmetric systems of linear equations*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 926–943.

[35] S. Karlin and L. S. Shapley, *Geometry of moment spaces*, Memorirs of the Americam Mathematical Society 12, Providence, (1953).

[36] C. Lanczos, *Solution of systems of linear equations by minimized iterations*, J. Res. Nat. Bureau Standards, 49 (1952), pp. 33–53.

[37] Matrix Market, `http://math.nist.gov/MatrixMarket/`. The Matrix Market is a service of the Mathematical and Computational Sciences Division of the Information Technology Laboratory of the National Institute of Standards and Technology.

[38] G. Meurant, *The computation of bounds for the norm of the error in the Conjugate Gradient algorithm*, Numerical Algorithms, 16 (1997), pp. 77–87.

[39] ——, *Numerical experiments in computing bounds for the norm of the error in the preconditioned Conjugate Gradient algorithm*, Numerical Algorithms 22, 3-4 (1999), pp. 353–365.

[40] N. Nachtigal, *A look-ahead variant of the Lanczos algorithm and its application to the quasi-minimal residual method for non-Hermitian linear systems*, PhD thesis, Massachusetts Institute of Technology, Cambrige, MA, 1991.

[41] C. C. Paige, *The Compution of Eigenvalues and Eigenvectors of Very Large Sparse Matrices*, PhD thesis, Intitute of Computer Science, University of London, London, U.K., 1971.

[42] ——, *Computational variants of the Lanczos method for the eigenproblem*, J. Inst. Maths. Applics, 10 (1972), pp. 373–381.

[43] ——, *Error analysis of the Lanczos algorithm for tridiagonalizing a symmetric matrix*, J. Inst. Maths. Applics, 18 (1976), pp. 341–349.

[44] ——, *Accuracy and effectiveness of the Lanczos algorithm for the symmetric eigenproblem*, Linear Algebra Appl., 34 (1980), pp. 235–258.

[45] C. C. Paige and Z. Strakoš, *Correspondence between exact arithmetic and finite precision behaviour of Krylov space methods*, in XIV. Householder Symposium, J. Varah, ed., U. of British Columbia, 1999, pp. 250–253.

[46] B. N. Parlett, *The Symmetric Eigenvalue Problem*, Prentice Hall, Englewood Cliffs, 1980.

[47] J. K. Reid, *On the method of conjugate gradient for the solution of large sparse systems of linear equations*, in Large Sparse Sets of Linear Equations, J. Reid, ed., N.Y., 1971, Academic Press.

[48] M. Rigal and J. Gaches, *On the compatibility of a given solution with the data of a given system*, J. Assoc. Comput. Mach., 14 (1967), pp. 543–548.

[49] H. G. Roos, M. Stynes, and L. Tobiska, *Numerical Methods for Singularly Perturbed Differential Equations*, vol. 24 of Springer Series in Computational Mathematics, Springer-Verlag, Berlin Heidelberg, 1996.

[50] Y. Saad, *Iterative Methods for Sparse Linear Systems*, PWS Publishing Company, 1996.

[51] P. E. Saylor and D. C. Smolarski, *Why Gaussian qudrature in the complex plane?*, Numerical Algorithms, 26 (2001), pp. 251–280.

[52] H. D. Simon, *Analysis of the symmetric Lanczos algorithm with reorthogonalization methods*, Linear Alg. Appl., 61 (1984), pp. 101–131.

[53] ——, *The Lanczos algorithm with partial reorthogonalization*, Math. Comput., 42 (1984), pp. 115–136.

[54] G. L. Sleijpen, H. A. van der Vorst, and D. R. Fokkema, *BiCGstab(l) and other hybrid BiCG methods*, Numerical Algorithms, 7 (1994), pp. 75–109.

[55] P. Sonneveld, *CGS, a fast Lanczos-type solver for nonsymetric linear systems*, SIAM J. Sci. Statist. Comput., 10 (1989), pp. 36–52.

[56] Z. Strakoš, *On the real convergence rate of the Conjugate Gradient method*, Lin. Alg. Appl., 154-156 (1991), pp. 535–549.

[57] ——, *Theory of Convergence and Effects of Finite Precision Arithmetic in Krylov Subspace Methods*, Thesis for the degree Doctor of Science, Institute of Computer Science, February 2001.

[58] Z. Strakoš and A. Greenbaum, *Open questions in the convergence analysis of the Lanczos process for the real symmetric eigenvalue problem*, ima preprint series 934, Univ. of Minnesota, 1992.

[59] Z. Strakoš and P. Tichý, *On error estimation in the Conjugate Gradient method and why it works in finite precision computations*, Submitted to ETNA, (2002), p. 26.

[60] G. Szegö, *Orthogonal Polynomials*, AMS Colloq. Publ. 23, AMS, Providence, 1939.

[61] P. Tichý, *Behaviour of BiCG and CGS algorithms*, master's thesis, Department of Numerical Mathematics, Faculty of Mathematics and Physics, Praha, 1997.

[62] ——, *The shadow vector in the Lanczos Method*, in Proceedings of the XUIIth summer school software and algorithms of numerical mathematics Nečtiny, 1999, pp. 309–320.

[63] P. Tichý and J. Zítko, *Derivation of BiCG from the conditions defining Lanczos' method for solving a system of linear equations*, Application of Mathematics 5, 43 (1998), pp. 381–388.

[64] H. A. van der Vorst, *BiCGSTAB: a fast and smoothly converging variant of BiCG for the solution of nonsymetric linear systems*, SIAM J. Sci. Statist. Comput., 13 (1993), pp. 327–341.

[65] H. F. Walker, *Implementation of the GMRES method using Hauseholder transformations*, SIAM Journal of Scientific Computing, 9 (1988), pp. 152–163.

[66] K. F. Warnick, *Nonincreasing error bound for the Biconjugate Gradient method*, report, University of Illinois, 2000.

[67] R. Weiss, *Convergence behavior of generalized conjugate gradient methods*, PhD thesis, University of Karlsruhe, 1990.