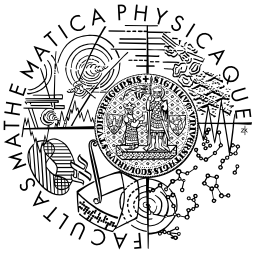


MATEMATICKO-FYZIKÁLNÍ FAKULTA
Univerzita Karlova

DIPLOMOVÁ PRÁCE

Jakub Kašpar

Propagace šumu v algoritmech kostruujících krylovovské regularizační báze pro řešení inverzních problémů

Katedra numerické matematiky

Vedoucí diplomové práce: doc. RNDr. Iveta Hnětynková, Ph.D.

Studijní program: Matematika pro informační technologie

Studijní obor: Matematika pro informační technologie

Praha 2023

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů. Tato práce nebyla využita k získání jiného nebo stejného titulu.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V Reykjavíku dne 25. 6. 2023 .....

Jakub Kašpar

Rád bych poděkoval vedoucí mé diplomové práce, doc. RNDr. Ivetě Hnětynkové, Ph.D., za všechen čas, který mi věnovala, za řadu cenných rad a za milou a trpělivou spolupráci.

Název práce: Propagace šumu v algoritmech kostruujících krylovovské regularizační báze pro řešení inverzních problémů

Autor: Jakub Kašpar

Katedra: Katedra numerické matematiky

Vedoucí diplomové práce: doc. RNDr. Iveta Hnětynková, Ph.D., Katedra numerické matematiky

Abstrakt: Tato práce se věnuje problému aproximace řešení lineárních inverzních úloh $Ax \approx b$ se zhlazujícím operátorem A a s pravou stranou b zanesenou náhodným šumem. Pro nalezení vhodné aproximace x lze využít celou třídu regularizačních metod, které iterativně odhadují řešení pomocí jeho projekce na vhodně zvolený Krylovův prostor malé dimenze. Navzdory tomu, že tato projekce má filtrační vlastnosti, dochází k postupné propagaci šumu do projekce, což vede k semikonvergenci metod. Znalost míry propagace šumu je pak zásadní pro nalezení nejpřesnější aproximace řešení.

Předložená práce studuje propagaci šumu v algoritmech Golub-Kahanovy iterační bidiagonalizace a Lanzosova algoritmu, které vytvářejí příslušný Krylovův prostor pro metody LSQR a MINRES. V práci analyzujeme koeficient, který v každé z metod amplifikuje šum, pro oba algoritmy je tento koeficient vyjádřen pomocí koeficientů Lanczosových polynomů, které jsou generovány při výpočtu ortonormální báze příslušného Krylovova prostoru. Pro Golub-Kahanovu iterační bidiagonalizaci jde o shrnutí a podrobnější rozepsání dostupné literatury, pro Lanczosův algoritmus jde o původní práci. Pro obě metody dále dokazujeme vztah mezi koeficientem amplifikujícím šum a normou příslušného rezidua. Teoretické poznatky z práce jsou ilustrovány numerickými experimenty využívajícími vlastní implementace příslušných metod. Experimentálně je studován i vliv propagace šumu na normu chyby hledaného řešení a chování popisovaných algoritmů v aritmetice s konečnou přesností.

Klíčová slova: inverzní problém, šum, regularizace, Krylovův prostor, ortogonální polynomy, amplifikační faktor

Title: Noise propagation in algorithms constructing Krylov regularization bases for the solution of inverse problems

Author: Jakub Kašpar

Department: Department of Numerical Mathematics

Supervisor: doc. RNDr. Iveta Hnětynková, Ph.D., Department of Numerical Mathematics

Abstract: In this thesis we consider a linear inverse problem $Ax \approx b$ with a smoothing operator A and a right-hand side vector b polluted by unknown noise. To find good approximation of x we can use large family of iterative regularization methods, which compute the approximate solution by projection onto a Krylov subspace of small dimension. Even though this projection has filtering property, the high frequency noise propagates to the Krylov basis, which causes semiconvergence of the methods. The knowledge of intensity of noise propagation is therefore necessary to find reasonably precise approximation of the solution.

In the thesis we study noise propagation in the Golub-Kahan iterative bidiagonalization and in the Lanczos algorithm, which construct the required Krylov subspace for LSQR and MINRES methods. For both methods, we analyze a noise-amplifying coefficient, for which we derive explicit formulas in both cases. For the Golub-Kahan bidiagonalization, this analysis summarizes the theory from multiple sources. Analysis for the Lanczos algorithm is original. For both methods, we derive explicit relations between noise-amplifying coefficients and residual norms. Several numerical experiments are presented to demonstrate properties of both algorithms. Impact of noise propagation on true errors and influence of finite-precision are also studied.

Keywords: inverse problem, noise, regularization, Krylov subspace, orthogonal polynomials, amplification factor

Obsah

Seznam použitého značení

$\mathbb{R}$	množina reálných čísel
$\mathbb{R}^n$	množina reálných sloupcových vektorů dimenze n
$\mathbb{R}^{m\times n}$	množina matic řádu $m \times n$ s reálnými koeficienty
$(\cdot;\cdot)_{L^2}$	L^2-skalární součin
$\|\cdot\|_2$	L^2-norma
$\|\cdot\|$	euklidovská norma
$\|\cdot\|_F$	Frobeniova norma
$.^T$	transpozice
A	matice lineárního problému
b	pravá strana lineárního problému
b^{exact}	pravá strana lineárního problému bez šumu
$Cov(\cdot)$	kovarianční matice
dim	dimenze vektorového prostoru
det	determinant
diag($\sigma_1,\ldots,\sigma_n$)	diagonální matice s čísly $\sigma_1,\ldots,\sigma_n$ na diagonále.
$\mathcal{D}_j$	množina všech posloupností čísel 1,2 se součtem j
e	vektor aditivního šumu
err_{rel}	relativní chyba
e_j	j-tý kanonický vektor
$f(t)$	řešení Frendholmovy integrální rovnice
$g(s)$	pravá strana Frendholmovy integrální rovnice
H_j	horní Hessenbergova matice
I	jednotková matice
j_{rev}	iterace, v níž dojde k projevení šumu
$K(s,t)$	jádro Frendholmovy integrální rovnice
$\mathcal{K}_j(A,b)$	j-tý Krylovův prostor asociovaný s A a b
L_j	dolní trojúhelníková bidiagonální matice řádu $j \times j$
L_{j+}	dolní trojúhelníková bidiagonální matice řádu $j+1 \times j$
$\mathcal{N}(0,\eta^2 I)$	normální rozdělení s nulovou střední hodnotou a směrodatnou odchylkou η
$\mathcal{P}_j$	prostor polynomů stupně j
r_j^{CRAIG}	reziduum metody CRAIG
r_j^{FOM}	reziduum metody FOM
r_j^{LSQR}	reziduum metody LSQR
r_j^{MINRES}	reziduum metody MINRES
RNL	relativní úroveň šumu
s_j	levý bidiagonalizační vektor
S_j	matice levých bidiagonalizačních vektorů
$\mathcal{S}_j$	permutační grupa na j prvcích.
span$\{\cdots\}$	lineární obal
U	matice levých singulárních vektorů
V	matice pravých singulárních vektorů

w_i	bázový vektor Krylovova prostoru $\mathcal{K}_j(A,b)$, potažmo $\mathcal{K}_j(A^TA,A^Tb)$
W_i	matice bázových vektorů w_i
x	řešení lineárního problému
x_j	přibližné řešení lineárního problému získané j-tou iterací dané numerické metody
x^{exact}	přesné řešení lineárního problému
x^{naive}	naivní řešení lineárního problému
y_j	řešení vnitřního problému v krylovovské metodě
α_i	diagonální prvky matic L_j a T_j
β_i	poddiagonální prvky matic L_j a T_j
$\{\Delta_\iota\}$	posloupnost z $\mathcal{D}_j$
$\varphi_j(0)$	koeficient, který násobí vektor šumu
Σ	diagonální matice singulárních čísel
σ_i	singulární čísla matice

Úvod

Inverzní úlohy jsou třídou matematických problémů, jejichž cílem je rekonstrukce neznámých vstupních dat na základě známého výstupu, který je však zanesen náhodným šumem. Potřeba řešení takovýchto úloh nastává v řadě různých situací – neznámými vstupními daty může být například původní, ostrý obrázek, zatímco známých výstupem může být pouze jeho rozmazaná a šumem zanesená podoba. Stejně tak může být neznámým vstupem i dvojrozměrný obrázek či trojrozměrný objekt, které máme nasnímány pomocí tomografu a musíme je ze získaných dat vykreslit. Podobné úlohy se vyskytují v geofyzice, seismologii, termodynamice i jinde [Han17],[Vog02],[Han10].

Vstupní data, proces, kterým procházejí i výstup, který můžeme měřit, bývají nezřídka spojité veličiny a procesy. Vzhledem k tomu, že naše měření jsou typicky diskrétní a že dané úlohy bývají často příliš složité pro analytické řešení, budeme se v celé této práci věnovat řešení diskrétních inverzních úloh – povětšinou získaných diskretizací spojitých úloh. Tím pádem budeme řešit lineární problémy tvaru

$$Ax \approx b, \quad A \in \mathbb{R}^{m\times n}, \quad b \in \mathbb{R}^m, \quad b = b^{exact} + e,$$

kde b^{exact} je přesným výstupem dané inverzní úlohy a e je aditivní náhodný šum. Naším cílem je nalézt co nejlepší aproximaci řešení x^{exact} takového, že

$$Ax^{exact} = b^{exact}.$$

Matice A bývá typicky velmi špatně podmíněná, což nám v kombinaci s náhodným šumem zaneseným do pravé strany b brání v nalezení přesného řešení. Z tohoto důvodu je nutné využít k řešení inverzních úloh nějakou formu regularizace. Klasickými regularizačními metodami je například Tichonovova regularizace [Tik63] a nebo TSVD [Han71]. Obě tyto metody vyjadřují řešení pomocí singulárního rozkladu matice, v němž potlačují složky ovlivněné šumem. Proto však potřebují alespoň částečný singulární rozklad matice A a jsou tedy vhodné především pro menší úlohy.

Vzhledem k tomu, že matice A bývá v praxi poměrně velká a často i řídká, bývají k řešení diskrétních inverzních úloh voleny krylovovské iterativní metody. Namísto přesného řešení x^{exact} v nich iterativně hledáme přibližná řešení x_j, která jsou aproximacemi vektoru x ve vhodně zvoleném projekčním podprostoru. Tento podprostor má výrazně menší dimenzi a zároveň je generován takovými vektory, které v řešení potlačí nepřesnosti způsobené šumem. Příklady takových metod jsou například metoda LSQR [PS82], z metody sdružených gradientů (CG, [HS52]) vycházející metody CGLS [BES98] a CGNE [Cra55] a nebo metoda minimálních reziduí (MINRES, [PS75]) a z ní vycházející metoda MR-II [Han95]. Oba přístupy je možné i kombinovat – v každé iteraci můžeme nejprve nalézt projekci problému do vhodného Krylovova podprostoru a v něm následně hledat řešení například s využitím Tichonovovy regularizace [Han10, 6.4], [CNO07].

Přirozeným problémem vyvstávajícím ve všech regularizačních metodách je míra, do jaké chceme řešení regularizovat – v klasických metodách jako Tichonovova regularizace nebo TSVD tuto míru určuje volba regularizačního parametru. V iterativních metodách je analogickým problémem volba iterace, v níž představuje x_j nejlepší možnou aproximaci x^{exact} a v níž je tedy vhodné metodu zastavit.

Od jisté iterace se totiž do projekčního prostoru a tedy i do x_k začne propagovat šum, který je do výpočtu zanesen vektorem b, s jehož pomocí vytváříme bázi Krylovova prostoru. Krylovovské metody proto vykazují semikonvergenci – před dosažením určité iterace získáváme postupně lepší a lepší aproximaci řešení, po dosažení této iterace šum postupně převládne i v aproximacích řešení.

V této práci se budeme věnovat především dvěma základním iterativním metodám: metodě LSQR a algoritmu Golub-Kahanovy bidiagonalizace [GK65], na kterém je postavena, a analogicky metodě MINRES postavené na Lanczosově algoritmu [Lan50]. V článku [HPS09] bylo ukázáno, jak se Golub-Kahanovou bidiagonalizací propaguje šum a následný článek [HKP17] ukázal vztah koeficientu, s nímž se tento šum zesiluje, s normou reziduí metod, které jsou na Golub-Kahanově bidiagonalizaci založené – mimo jiné i metody LSQR.

Tato diplomová práce si klade za cíl shrnout nezbytnou základní teorii nutnou ke studiu diskrétních inverzních úloh a s jejím využitím podrobněji rozepsat analýzu šíření šumu v Golub-Kahanově bidiagonalizaci a vyjádření rezidua metody LSQR, provedenou v [HPS09] a [HKP17]. Následně pak podobnou analýzu provedeme i pro Lanczosův algoritmus, metodu MINRES a metody z ní vycházející. Dokážeme explicitní formuli pro koeficient, s nímž se v bázových vektorech generovaných Lanczosovým algoritmem šum propaguje, a odvodíme vztah mezi tímto koeficientem a normou rezidua metody MINRES. Koeficient amplifikující šum explicitně popíšeme i pro Arnoldiho algoritmus, který představuje zobecnění Lanczosova algoritmu a stojí v základu metody GMRES [SS86].

Samotná práce sestává ze čtyř kapitol: v první kapitole shrnujeme základní teoretické poznatky, z nichž vycházejí krylovovské regularizační metody, hlavními zdroji byly přehledové publikace [Han10] a [DTHPS12]. Druhá kapitola představuje analýzu Golub-Kahanovy iterační bidiagonalizace a metody LSQR. Tato analýza sleduje články [HPS09] a [HKP17], které jsou doplněny vlastními podrobnějšími důkazy klíčových tvrzení. Třetí kapitola této práce provádí analogickou analýzu pro metody založené na Lanczosově algoritmu a dospívá k vyjádření rezidua metody MINRES. V poslední sekci je první krok této analýzy proveden i pro Arnoldiho algoritmus. Veškerá analýza z této kapitoly je původní prací autora.

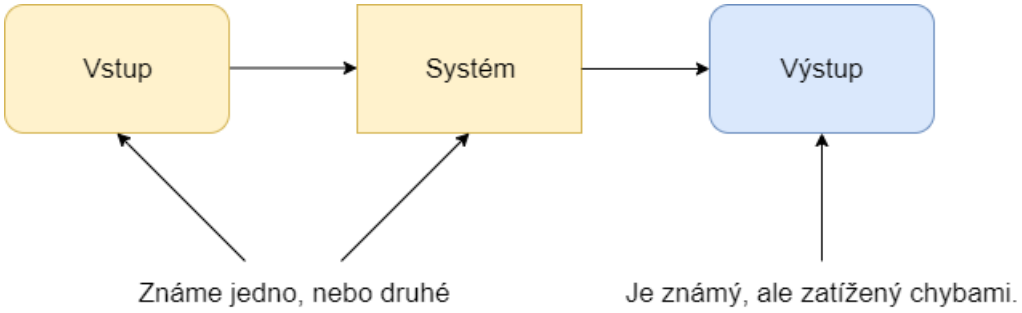
Poslední kapitola obsahuje numerické experimenty studující některé fenomény z Kapitol 2 a 3, pro které autorovi zatím není známo rigorózní matematické zdůvodnění. Numerickými experimenty jsou zároveň i ilustrovány poznatky ze všech tří předchozích kapitol. Využíváme zde knihovnu Regularization Tools [Han07], z níž čerpáme testovací úlohy a některé skripty. Implementace zkoumaných numerických metod jsou původním dílem autora vycházejícím z [Han10] a [DTHPS12].

1. Inverzní úlohy

V této kapitole shrneme základní teorii o inverzních úlohách a Krylovových prostorech, již budeme posléze využívat v jednotlivých metodách a jejich analýze. Základním zdrojem, který tuto teorii shrnuje, je [Han10], u jednotlivých poznatků budeme vycházet i z dalších, úžeji zaměřených zdrojů.

1.1 Úvod do spojitých inverzních úloh

Metody, které tato práce studuje, cílí na řešení specifické třídy matematických problémů - tzv. *inverzních úloh*. Předpokládejme, že máme systém, který danému vstupu přiřadí nějaký výstup. Zadání inverzní úlohy může být dvou druhů:



Obrázek 1.1: Schéma inverzní úlohy – vstup nebo systém známe, výstup známe také, ale typicky je zatížen šumem.

Řešení inverzních úloh je často velmi citlivé na nepřesnosti v naměřených datech – říkáme, že inverzní úlohy jsou *špatně postavené* (ill-posed). Tento pojem vymezil Jacques Hadamard ve svém článku [Had02] takto:

Definice 1. *Matematický problém řešící fyzikální úlohu je dobře postavený, pokud pro něj existuje jednoznačné řešení, které se mění spojitě v závislosti na naměřených datech. V opačném případě řekneme, že problém je špatně postavený.*

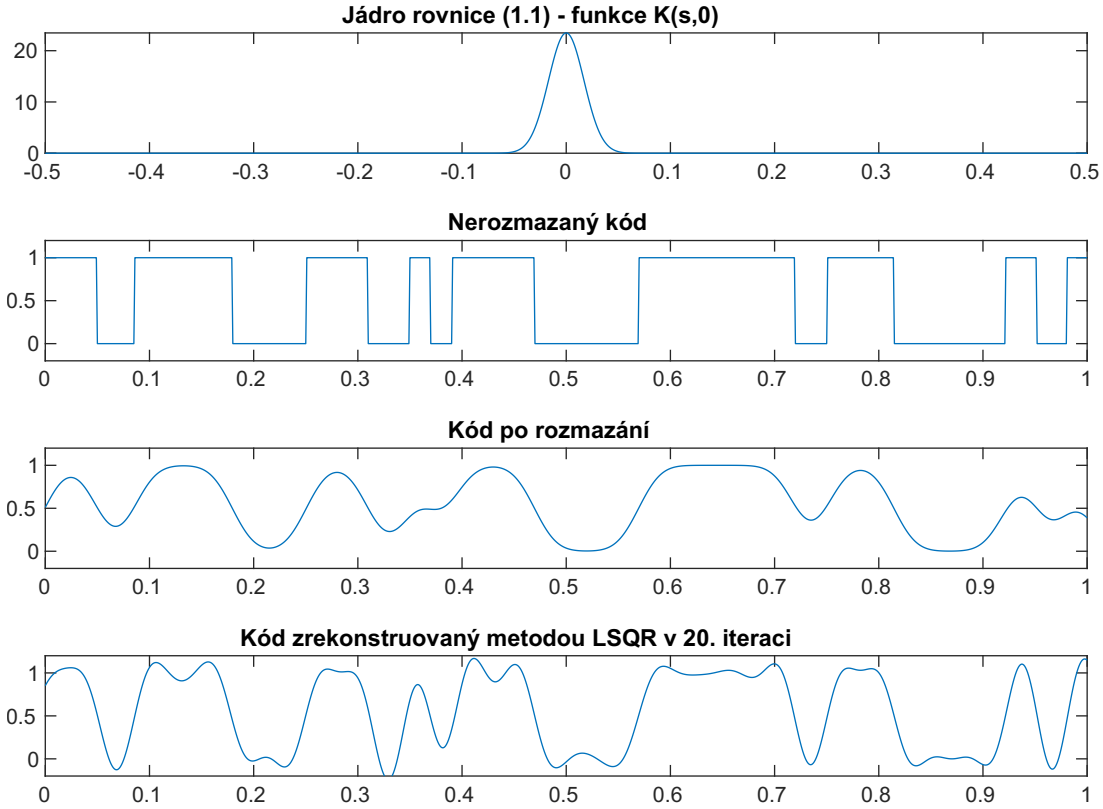
Vzhledem k tomu, že budeme pracovat pouze s lineárními modely diskretizovaných problémů, Hadamardovu definici upravíme: úloha je špatně postavená, nejen když se její řešení nemění spojitě v závislosti na datech, ale i když drobná změna našich dat vede k zásadně odlišnému řešení.

Tato práce bude pracovat s algoritmy řešícími třídu inverzních úloh, které mají tvar *Fredholmovy integrální rovnice 1. druhu*:

$$g(s) = \int_0^1 K(s,t)f(t)dt, 0 \leq s \leq 1. \tag{1.1}$$

Funkce $K(s,t)$ a $g(s)$ známe, funkce $f(t)$ je hledaná neznámá. Funkci $K(s,t)$ nazýváme *jádro* Fredholmovy rovnice a představuje systém, přiřazující neznámým vstupním datům $f(t)$ naměřená data $g(s)$. Tyto úlohy vyvstávají v celé řadě aplikací. Detaily o těchto problémech najde čtenář v [Kre99, 15].

Typickým příkladem inverzní úlohy je například zaostření obrazu – pro jednoduchost ukážeme úlohu s jednorozměrným obrázkem, konkrétně čárovým kódem,



Obrázek 1.2: Příklad rozmazání a doostření čárového kódu. Rozmazání je modelováno funkcí $K(s,t)$ definovanou v rovnici (1.2) s koeficientem $\mu = 0{,}017$.

který představuje vstupní měření $f(t)$. Čárový kód je sice ostrý, při čtení je však rozmazán. Toto rozmazání můžeme modelovat pomocí gaussovského šumu [Pra78, 4.3.2], tj. jako dosazení $f(t)$ do rovnice (1.1), kde jádro K je rovno

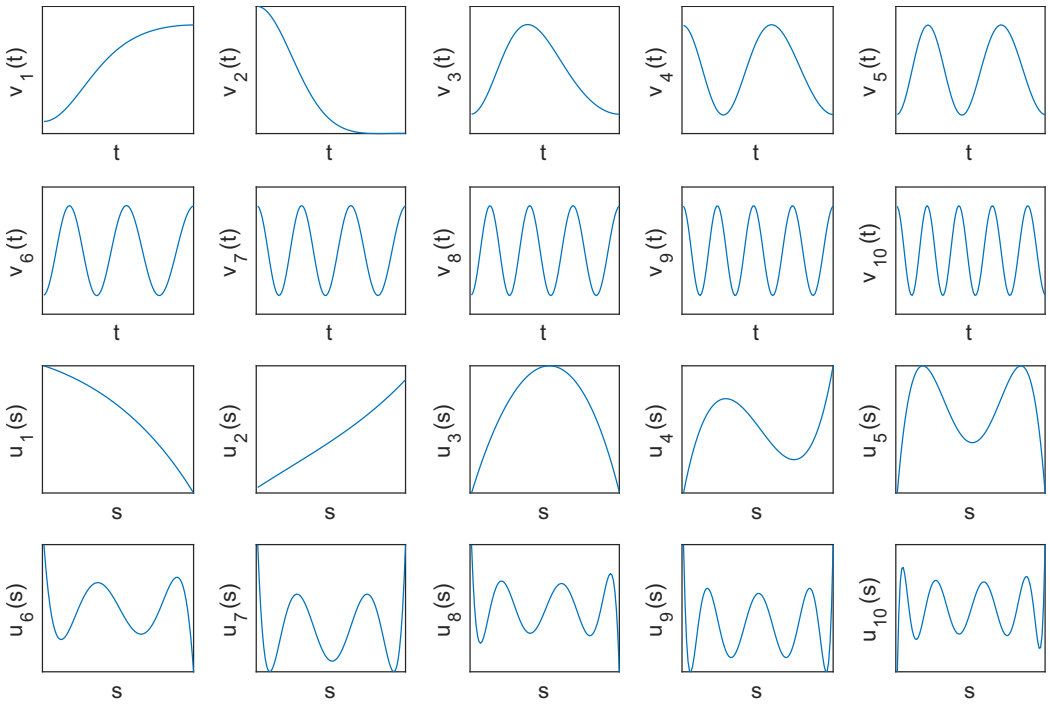
$$K(s,t) = \frac{1}{\mu\sqrt{2\pi}} e^{\frac{-(s-t)^2}{2\mu^2}}, \tag{1.2}$$

kde koeficient μ ovlivňuje míru rozmazání (čím větší μ, tím menší rozmazání). Příklad takového čárového kódu, jeho rozmazání a následného doostření za použití metody LSQR (popsanou v Kapitole 2.3) ukazuje Obrázek 1.2. Zatím nepředpokládáme zanesení výsledného rozmazaného kódu $g(s)$ aditivním šumem.

Klíčovou vlastností Fredholmovy integrální rovnice je následující skutečnost.

Věta 1. *Definujme posloupnosti funkcí $f_p(t) = \sin(2\pi pt), p \in \mathbb{N}$. Dále definujme $g_p(s) = \int_0^1 K(s,t) f_p(t) dt$. Pak $\lim_{p\to\infty} g_p(s) = 0$.*

Tato věta je důsledkem Riemann-Lebesgueova lemmatu (viz [Gol76, Theorem 12.5C]) a plyne z ní, že vyšší frekvence ve funkci f jsou v modelu (1.1) utlumeny, a funkce $g(s)$ je proto oproti funkci $f(t)$ výrazně hladší. To má pro řešení inverzních úloh fundamentální důsledky: ve funkci f, kterou hledáme, budou oproti funkci g výrazně posílené vyšší frekvence. I malá perturbace funkce g může tedy vést k libovolně velké změně hledané funkce f – stačí, když v sobě tato perturbace ponese i vyšší frekvence.



Obrázek 1.3: Singulární funkce $u_i(s)$ a $v_i(t)$ pro modelovou úlohu `baart(100)`.

Pro studium vlastností diskrétních inverzních úloh je klíčová následující věta:

Věta 2 (Singulární rozvoj)**.** *Předpokládejme, že* $\int_0^1 \int_0^1 K(s,t)dtds < \infty$. *Pak můžeme funkci* $K(s,t)$ *rozvinout do řady*

$$K(s,t) = \sum_{i=1}^{\infty} \mu_i u_i(s) v_i(t).$$

Skaláry μ_i *nazveme singulárními hodnotami a funkce* u_i *a* v_i *levými a pravými singulárními funkcemi. Navíc* $\int_0^1 u_i(s)u_j(s) = \int_0^1 v_i(t)v_j(t) = \delta_{ij}$ *a* $\mu_1 \geq \mu_2 \geq \ldots$, *přičemž* $\mu_i \geq 0$ *pro všechna* $i \in \mathbb{N}$.

Vidíme tedy, že jádro $K(s,t)$ můžeme rozepsat jako součet nekonečné řady, v níž koeficienty μ_i určují dopad $K(s,t)$ na singulární funkce $u_i(s)$ a $v_i(t)$. Stojí za povšimnutí, že čím vyšší i, tím mají singulární funkce u_i a v_i vyšší frekvenci.

Věta 3. *Pro všechna* $i \in \mathbb{N}$ *platí*

$$\int_0^1 K(s,t)v_i(t)dt = \mu_i u_i(s). \tag{1.3}$$

Levé i pravé singulární funkce tvoří báze prostoru $L_2[0,1]$, proto můžeme funkce f a g rozvinout do bází definovaných singulárními funkcemi:

$$f(t) = \sum_{i=1}^{\infty} (v_i,f)_{L_2} v_i(t), \qquad g(s) = \sum_{i=1}^{\infty} (u_i,g)_{L_2} u_i(s).$$

Dosadíme-li tyto rozvoje do rovnice (1.3), dostaneme rovnost

$$\sum_{i=1}^{\infty} (u_i,g)_{L^2} u_i(s) = g(s) = \int_0^1 K(s,t)f(t)dt = \sum_{i=1}^{\infty} \mu_i (v_i,f)_{L^2} u_i(s),$$

a díky ortogonalitě u_i proto platí $(u_i,g)_{L^2} = \mu_i(v_i,f)_{L^2}$ pro všechna přirozená i. Za předpokladu, že pro všechna $i \in \mathbb{N}$ je $\mu_i > 0$, můžeme tento vztah dosadit zpět do rozvoje f. Dostaneme vyjádření

$$f(t) = \sum_{i=1}^{\infty} \frac{(u_i,g)_{L_2}}{\mu_i} v_i(t). \tag{1.4}$$

K další analýze singulárního rozvoje budeme potřebovat, aby koeficienty $(u_i,g)_{L_2}$ klesaly rychleji než μ_i. Formálně to garantuje následující podmínka.

Definice 2 (Picardova podmínka)**.** *Řekneme, že inverzní úloha splňuje Picardovu podmínku, jestliže*

$$\sum_{i=1}^{\infty} \left(\frac{(u_i,g)_{L_2}}{\mu_i} \right) < \infty.$$

Cenným důsledkem rovnosti (1.4) je následující věta.

Věta 4. *Pokud funkce f splňuje Picardovu podmínku, pak je kvadraticky integrovatelná, tj. $\|f\|_2^2 = \int_0^1 f(x)^2 dx < \infty$.*

Důkaz.

$$\|f\|_2^2 = \int_0^1 f(t)^2 dt = \sum_{i=1}^{\infty} (v_i,f)_{L_2} = \sum_{i=1}^{\infty} \left(\frac{(u_i,g)_{L_2}}{\mu_i} \right) < \infty.$$

□

Může se zdát, že podmínka kvadratické integrovatelnosti není prakticky příliš užitečná, v praxi však chceme, aby funkce f řešící rovnici (1.1) byla hladká, a tedy aby v její Fourierově transformaci byly výrazně silnější nižší frekvence. Pro jednoduchost proto dále můžeme předpokládat, že pokud platí Picardova podmínka, je naše hledané řešení hladké. Problém spočívá v tom, že ač přesná pravá strana g Picardovu podmínku splňuje, ve skutečnosti je funkce g naměřena nepřesně a zatížena šumem. Řada $\sum_{i=1}^{\infty} \left(\frac{(u_i,g)_{L_2}}{\mu_i} \right)$ proto může divergovat, což je podrobně vysvětleno na diskrétním modelu v Sekci 1.3. Podrobnější teoretické poznatky o singulárním rozvoji lze najít mimo [Han10, 3] i v [Kre99, 15.4.].

1.2 Diskretizace inverzních úloh

Pro rovnici (1.1) typicky není v našich silách nalézt analytické řešení. K nalezení alespoň přibližného řešení je tudíž nutné problém převést na diskrétní – z původní úlohy tím vytvoříme lineární problém $Ax \approx b$, kde $A \in \mathbb{R}^{m \times n}$ odpovídá jádru K, hledaný vektor $x \in \mathbb{R}^n$ odpovídá funkci f a $b \in \mathbb{R}^m$ odpovídá funkci g (samozřejmě s určitou diskretizační chybou). Stručně zde proto přiblížíme dva základní přístupy k diskretizaci integrální rovnice – podrobnosti lze nalézt např. v [Bak77] nebo [DM85].

Kvadratury

Rovnici (1.1) můžeme jednoduše numericky zintegrovat pomocí některé z kvadraturních metod. Pak

$$\int_0^1 K(s,t)f(t)dt \approx \sum_{j=1}^{n} \omega_j K(s,t_j)f(t_j),$$

kde t_j jsou zadané uzly a ω_j jim příslušné váhy. I tato aproximace je stále spojitou funkcí závislou na s. Data g máme typicky naměřená v konkrétních bodech s_i, proto budeme předpokládat, že

$$g(s_i) \approx \sum_{j=1}^{n} \omega_j K(s_i,t_j)f(t_j), i = 1,\ldots,m.$$

Položíme-li nyní $a_{ij} \equiv \omega_j K(s_i,t_j), x_j \equiv f(t_j)$ a $b_i \equiv g(s_i)$, vytvoříme z rovnice (1.1) lineární problém tvaru $Ax = b$. Tento přístup je výpočetně velmi jednoduchý, jeho přesnost je pak závislá zejména na volbě konkrétní kvadratury a její chybě. Metoda s sebou však nese jistá úskalí: především nevíme, v jakém vztahu bude singulární rozvoj K a singulární rozklad takto získané matice A.

Galerkinova metoda

Fundamentálně odlišným přístupem je vyjádření funkcí, s nimiž pracujeme, jako prvků v prostoru funkcí s nějakou příznivou bází. Klasickou metodou, která takto funguje, je Galerkinova metoda, jejíž podstatu zde nyní vysvětlíme. Základní myšlenkou je, že funkce $f(t)$ a $g(s)$ aproximujeme jako

$$f(t) \approx f_n(t) = \sum_{j=1}^{n} x_j \phi_j(t), g(s) \approx g_m(s) = \sum_{i=1}^{m} \xi_i \psi_i(s).$$

kde $\{\phi_j\}_{j=1}^n$ a $\{\psi_i\}_{i=1}^m$ jsou lineárně nezávislé posloupnost funkcí a

$$(f - f_n,\phi_k)_{L_2} = (g - g_m,\phi_l)_{L_2} = 0 \quad \text{pro} \quad k = 1,\ldots,n,\ l = 1,\ldots,m.$$

Dosazení aproximace f_n namísto f do rovnice (1.1) nám dá rovnost

$$g(s) \approx \sum_{j=1}^{n} \int_0^1 K(s,t)x_j\phi_j(t)dt.$$

Protože $(g - g_m,\psi_i)_{L_2} = 0$ pro $i = 1,\ldots,m$, platí

$$(g,\psi_i)_{L_2} = (g_m,\psi_i)_{L_2} \approx \Big(\sum_{j=1}^{n} \int_0^1 K(s,t)x_j\phi_j(t)dt,\psi_i\Big)_{L_2},$$

a z linearity skalárního součinu proto

$$(g,\psi_i)_{L_2} = (g_m,\psi_i)_{L_2} \approx \sum_{j=1}^{n} x_j \Big(\int_0^1 K(s,t)\phi_j(t)dt,\psi_i\Big)_{L_2}.$$

Nyní tedy můžeme položit

$$\begin{aligned} b_i &\equiv (g,\psi_i)_{L_2} = \int_0^1 g(s)\psi_i(s)ds, \\ a_{ij} &\equiv \Big(\int_0^1 K(s,t)\phi_j(t)dt,\psi_i \Big)_{L_2} = \int_0^1 \int_0^1 K(s,t)\phi_j(t)\psi_i(s)dsdt. \end{aligned} \tag{1.5}$$

Není-li v našich silách spočítat integrály z (1.5) analyticky, bývají vypočteny numericky pomocí vhodné kvadratury.

Oproti kvadraturám s sebou Galerkinova metoda nese zásadní výhodu pro numerickou analýzu, neboť (jak ukázal [Han88]) se řada vlastností operátoru K přenáší i do matice A. Praktickou výhodou pak je, že konkrétní volbou bázových funkcí $\{\varphi_j\}$ můžeme vynutit konkrétní vlastnosti řešení – víme-li kupříkladu, že funkce f je nezáporná a nebo rostoucí, můžeme i funkce φ_j volit nezáporné a nebo rostoucí. Slabinou Galerkinovy metody je výrazně vyšší výpočetní náročnost.

1.3 Vlastnosti diskrétních inverzních úloh

Počínaje touto kapitolou budeme v celé práci (v souladu s praxí) předpokládat, že matice $A \in \mathbb{R}^{m\times n}$, kde $m \geq n$.

Podobně jako můžeme jádro K rozvinout do nekonečné řady pomocí singulárního rozvoje, matici A můžeme zapsat pomocí *singulárního rozkladu* (SVD)

$$A = U\Sigma V^T = \sum_{i=1}^{n} u_i \sigma_i v_i^T,$$

kde matice $\Sigma \in \mathbb{R}^{m\times n}$ je diagonální matice se *singulárními čísly* na diagonále, pro kterou platí

$$\Sigma = \mathrm{diag}(\sigma_1,\ldots,\sigma_n), \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0,$$

a matice $U \in \mathbb{R}^{m\times m}$ a $V \in \mathbb{R}^{n\times n}$ jsou ortogonální matice, jejichž sloupce nazveme *levé,* resp. *pravé singulární vektory.*

Protože u_i a v_i tvoří ortonormální báze, můžeme s jejich pomocí zapsat

$$x = \sum_{i=1}^{n} (v_i^T x) v_i, \qquad\qquad b = \sum_{i=1}^{m} (u_i^T b) u_i.$$

Vyjádření x nám s využitím singulárního rozkladu dává rovnici

$$Ax = \sum_{i=1}^{m} \sigma_i (v_i^T x) u_i.$$

Srovnáme-li nyní koeficienty v rozvojích Ax a b, dostaneme rovnost $\sigma_i(v_i^T x) = u_i^T b$. Předpokládejme nyní na chvilku, že matice $A \in \mathbb{R}^{n\times n}$ je regulární čtvercová – pak $\sigma_i > 0$ pro $i = 1,\ldots n$. Z naší rovnice v takovém můžeme vyjádřit řešení x analogicky, jako jsme vyjadřovali f v rovnici (1.4):

$$x = \sum_{i=1}^{n} \frac{u_i^T b}{\sigma_i} v_i = V\Sigma^{-1}U^T b = A^{-1}b. \tag{1.6}$$

Vzhledem k tomu, že σ_i mohou být velmi blízká 0 a součiny $u_i^T b$ mohou být zaneseny šumem, není tento vzorec vhodný k praktickému použití. Toto vyjádření x budeme dále značit x^{naive} a budeme jej nazývat *naivní řešení.*

Vytvoříme-li matici $A \in \mathbb{R}^{n\times n}$ diskretizací jádra K pomocí Galerkinovy metody, [Han88] ukazuje, že její singulární hodnoty $\sigma_i^{(n)}$ budou aproximovat μ_i ze singulárního rozvoje K. Přesněji, pokud definujeme

$$\delta_n = \sqrt{\int_0^1 \int_0^1 K(s,t)dsdt - \|A\|_F},$$

pak budou pro $i = 1,2,\ldots,n$ platit následující rovnosti:

$$0 \leq \mu_i - \sigma_i^{(n)} \leq \delta_n,$$

$$\sigma_i^{(n)} \leq \sigma_i^{(n+1)} \leq \mu_i.$$

Podobně můžeme aproximovat i levé a pravé singulární funkce K. Levé singulární funkce jsou aproximovány funkcemi

$$u_j^{(n)}(s) = \sum_{i=1}^{n} u_{ij}\psi_i(s),$$

pravé singulární funkce pak funkcemi

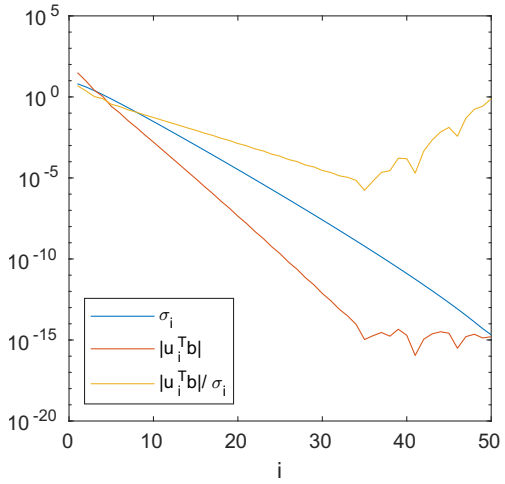
$$v_j^{(n)}(t) = \sum_{i=1}^{n} u_{ij}\phi_i(t).$$

Podmínka čtvercové regulární matice A se může zdát omezující, v praxi však bývá volba stejného počtu bázových funkcí pro f a g běžná. Analogické vztahy navíc platí i tehdy, když je matice A vzniklá diskretizací obdélníková, jedinou podmínkou, kterou [Han88] uvádí, je $n \neq m$[6]. V důsledku všech těchto vztahů mají spojité inverzní úlohy a úlohy vzniklé jejich diskretizací stejné vlastnosti. Singulární vektory u_i a v_i odpovídají příslušným singulárním funkcím do té míry, že i pro ně platí, že čím vyšší i, tím jsou ve Fourierově transformaci u_i a v_i silnější vyšší frekvence. V analogii k Picardově podmínce proto můžeme definovat následující podmínku, kterou navrhl [Han90]:

Definice 3 (Diskrétní Picardova podmínka)**.** *Řekneme, že pravá strana b lineárního problému $Ax = b$ splňuje diskrétní Picardovu podmínku, jestliže hodnoty $|u_i^T b|$ klesají v průměru rychleji než σ_i.*

Vzhledem k tomu, že v reálu pracujeme v konečné aritmetice, dává smysl uvažovat pouze $|u_i^T b|$ a σ_i větší než strojová přesnost. Obrázek 1.4 ilustruje hodnoty těchto veličin v jedné z klasických modelových úloh. Tato úloha splňuje diskrétní Picardovu podmínku, což ilustruje klesající podíl $|u_i^T b|/\sigma_i$. Od $i = 35$ se koeficienty $|u_i^T b|$ dostávají na hranici strojové přesnosti (10^{-14}), jejich další pokles již proto nevidíme. Můžeme si všimnout, že na hranici strojové přesnosti je i σ_{50}.

Pokud původní data splňují Picardovu podmínku, můžeme očekávat, že diskretizovaná varianta bude splňovat diskrérní Picardovu podmínku. Důsledkem podobnosti mezi spojitým a diskrétním problémem je, že podobně jako jádro K bude i matice A zhlazovat vstupní vektory x – srovnáme-li diskrétní Fourierovu transformaci x a Ax, bude mít Ax potlačené vyšší frekvence.



Obrázek 1.4: Hodnoty σ_i,$|u_i^T b|$ a jejich podíl pro modelovou úlohu `gravity(50)`.

1.4 Šum a jeho role v řešení diskrétních inverzních úloh

V této práci se pro jednoduchost omezíme na situace, kde je šumem zatížena pouze[1] pravá strana rovnice b. Pokud zanedbáme diskretizační chyby, můžeme předpokládat, že existují x^{exact} a b^{exact} taková, že platí rovnice $Ax^{exact} = b^{exact}$. Problém zatížený šumem budeme nyní modelovat tak, že namísto b^{exact} budeme pracovat s b, kde

$$b = b^{exact} + e,$$

přičemž e reprezentuje vektor aditivního šumu. Typicky je $\|e\|$ výrazně menší než $\|b^{exact}\|$.

Definice 4. *Jako relativní hladinu šumu označíme podíl*

$$RNL = \frac{\|e\|}{\|b^{exact}\|}.$$

V této práci budeme typicky pracovat s *bílým šumem*, tj. s vektorem e, jehož složky nejsou korelovány – $Cov(e) = \eta^2 I$. Dále budeme předpokládat, že šum není závislý na datech – složky b^{exact} a e nejsou korelovány. Nepřesností měření typicky vzniká *Gaussovský šum* – vektor šumu e splňuje

$$e \sim \mathcal{N}(0,\eta^2 I).$$

Pokud v rovnici (1.6) dosadíme $b = b^{exact} + e$, dostaneme rovnici

$$x^{naive} = A^{-1}b = A^{-1}(b^{exact} + e) = x + A^{-1}e.$$

Kovarianční matice x^{naive} pak bude

$$Cov(x^{naive}) = Cov(A^{-1}b) = Cov(A^{-1}e) = A^{-1}Cov(A)^{-T} = \eta^2(AA^T)^{-1}$$

[1] V praxi je situace pochopitelně složitější, ale zatímco chyby v A jsou typicky dány přílišným zjednodušením našeho modelu a jeho příliš hrubou diskretizací, vektor b je typicky produktem nějakého nepřesného měření.

a její norma

$$\|Cov(x^{naive})\|_2 = \eta^2 \|(AA^T)^{-1}\|_2 = \frac{\eta^2}{\sigma_n},$$

kde σ_n je nejmenší singulární hodnota matice A. Protože A bývá špatně podmíněná, celá kovarianční matice má vysokou normu, v absolutní hodnotě jsou proto velké i její prvky, a v důsledku toho se řešení x^{naive} stává nepoužitelným.

Dosadíme-li naši podobu b do součinu $U^T b$, budou jeho složky mít podobu

$$u_i^T b = u_i^T b^{exact} + u_i^T e.$$

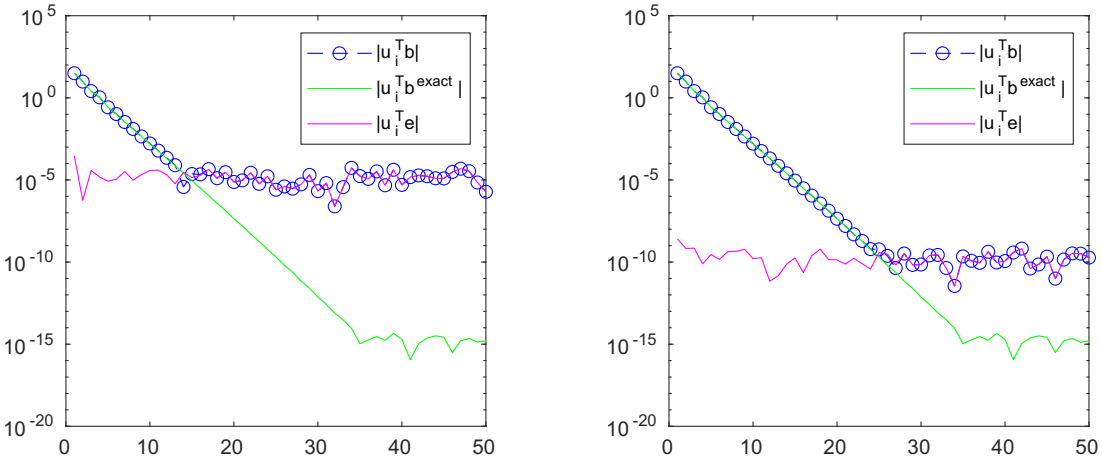
Zároveň

$$Cov(U^T e) = U^T Cov(e) U = \eta^2 U^T U = \eta^2 I,$$

vektor $U^T e$ je tedy náhodný vektor, jehož složky mají stejný rozptyl jako složky vektoru e. Původní vektor b^{exact} splňoval diskrétní Picardovu podmínku, hodnoty $|u_i^T b^{exact}|$ proto klesaly rychleji než σ_i. Hodnoty $|u_i^T e|$ jsou náhodně rozdělené a neklesají – tím pádem musí existovat nějaké κ takové, že

- pro $i < \kappa$ je $|u_i^T b^{exact}| > |u_i^T e|$, proto $u_i^T b \approx u_i^T b^{exact}$, a součiny $u_i^T b$ tedy nesou informaci o b^{exact},
- pro $i > \kappa$ je $|u_i^T b^{exact}| < |u_i^T e|$, proto $u_i^T b \approx u_i^T e$ a součiny $u_i^T b$ nenesou žádnou relevantní informaci – jsou tvořeny především šumem.

Situaci ilustruje následující obrázek: grafy představují koeficienty $|u_i^T b|$,$|u_i^T b^{exact}|$ a $|u_i^T e|$ pro pravou stranu b z modelové úlohy z Obrázku 1.4 zatíženou dvěma různými úrovněmi šumu.



(a) $RNL = 10^{-5}$ *(b)* $RNL = 10^{-10}$

Obrázek 1.5: Koeficienty $|u_i^T b|$,$|u_i^T b^{exact}|$ a $|u_i^T e|$ pro úlohu `gravity(50)` *s pravou stranou b zatíženou šumem o různých hladinách.*

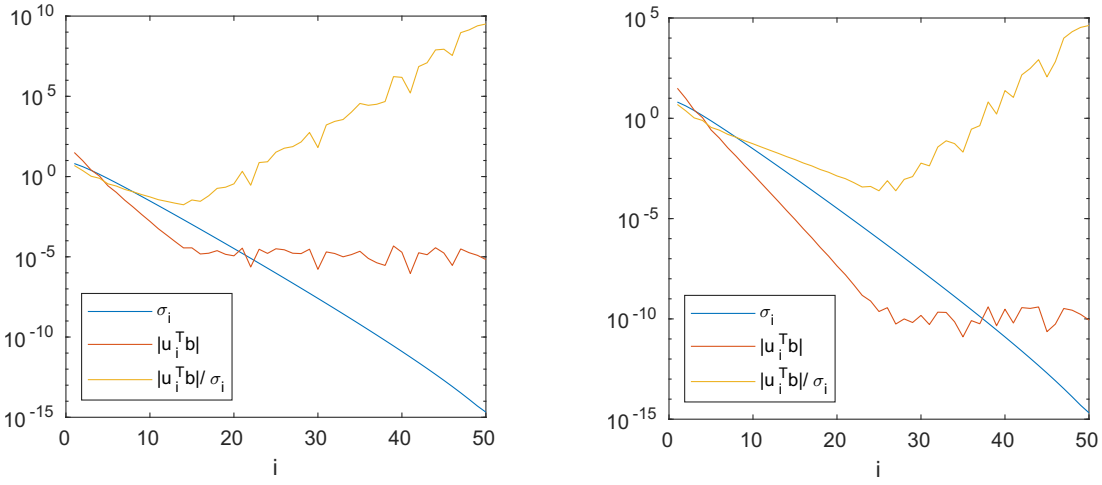
Budeme-li nyní hledat řešení x pomocí rovnice (1.6), narazíme na to, že pro $i > \kappa$ bude

$$\frac{u_i^T b}{\sigma_i} v_i \approx \frac{u_i^T e}{\sigma_i} v_i. \tag{1.7}$$

Vzhledem k tomu, že vektor e je náhodný a nesplňuje diskrétní Picardovu podmínku, $|u_i^T e|$ neklesá tak rychle jako σ_i a koeficienty u vektorů v_i s vyšší frekvencí

jsou tedy čím dál větší. V důsledku toho toto vyjádření x nemůžeme použít, bude totiž zaneseno vysokými frekvencemi z vektorů u_i pro $i > \kappa$, které jsou přenásobeny pomalu klesajícími koeficienty e a vyděleny σ_i blížícími se k nule.

Celou tuto situaci ilustruje přiložený Obrázek 1.6. Pravá strana modelové úlohy z Obrázku 1.4 je v něm zatížena šumem o dvou různých hladinách šumu a vidíme, že je tím pádem porušena diskrétní Picardova podmínka – hodnoty $|u_i^T b|$ přestávají po dosažení úrovně šumu klesat, protože v nich převažuje $|u_i^T e|$ nad $|u_i^T b^{exact}|$.



(a) $RNL = 10^{-5}$ *(b)* $RNL = 10^{-10}$

Obrázek 1.6: Hodnoty σ_i,$|u_i^T b|$ *a jejich podíl pro úlohu* `gravity(50)` *s pravou stranou* b *zatíženou šumem o různých hladinách.*

1.5 Projekční metody a Krylovovy prostory

V dalších kapitolách této práce budeme analyzovat *projekční metody* řešení inverzních úloh, a to v *Krylovových prostorech*. Teorie zde popsaná vychází zejména z [DTHPS12] a [Han10]. Princip projekčních metod je jednoduchý: namísto hledání přesného x v celém $\mathbb{R}^n$, při kterém narážíme na špatně podmíněnou úlohu a šumem zanesený vektor b, hledáme řešení $\tilde{x}$ ve vhodném menším podprostoru. To s sebou nese dvě výhody:

- výrazně se tím zmenší dimenze problému, který řešíme,
- podprostor můžeme volit tak, aby byl generován pouze hladkými vektory – tím pádem řešení z něj nebude zaneseno vysokofrekvenčním šumem.

Řečeno matematicky: namísto x hledáme

$$\tilde{x} = \text{argmin}_{\hat{x}} \|A\hat{x} - b\|_2,\ \hat{x} \in \mathcal{D}_k = \text{span}\{d_1, \ldots, d_k\},$$

kde zpravidla $k \ll n$. Tuto podmínku můžeme formulovat jako řešení mnohem menší soustavy lineárních rovnic – označme D_k matici se sloupci $d_1, \ldots, d_k$. Pak

$$\tilde{x} = D_k \tilde{y}, \quad \tilde{y} = \text{argmin}_y \|AD_k y - b\|_2. \tag{1.8}$$

Volba podprostoru $\mathcal{D}_k$ se liší metodu od metody – s ohledem na vztahy (1.6) a (1.7) se zdá jako ideální volba span$\{v_1,\ldots,v_i\}$ pro $i < \kappa$. Háček je v tom, že pro velkou matici A není v našich silách počítat singulární rozklad. Z tohoto důvodu použijeme jiný podprostor – konkrétně budeme pracovat s Krylovovými prostory. Pro jejich konstrukci nám totiž stačí pro danou matici A opakovaně počítat $A\vartheta$ nebo $A^T\varepsilon$ pro vhodné vektory ϑ a ε.

Definice 5 (Krylovův prostor)**.** *Mějme dánu čtvercovou matici $M \in \mathbb{R}^{n\times n}$ a vektor $\phi \in \mathbb{R}^n$. Pak k-tý Krylovův prostor asociovaný s M a ϕ definujeme jako*

$$\mathcal{K}_k(M,\phi) = \text{span}\{\phi, M\phi, M^2\phi, \ldots, M^{k-1}\phi\}$$

Vidíme, že $\dim\mathcal{K}_k(M,\phi) \leq \min\{k,n\}$, a že

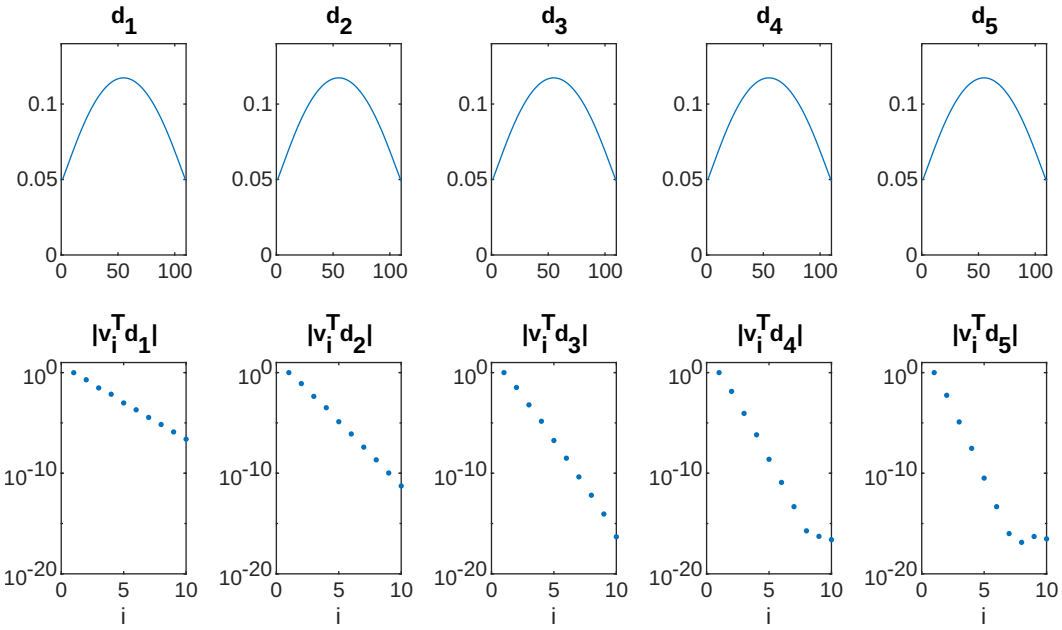
$$\mathcal{K}_1(M,\phi) \subseteq \mathcal{K}_2(M,\phi) \subseteq \cdots \subseteq \mathcal{K}_l(M,\phi) = \mathcal{K}_{l+1}(M,\phi)$$

pro nějaké $l \leq n$. Toto l nazveme *stupeň vektoru ϕ vzhledem k matici M* a dále jej budeme značit $st_\phi(M)$.

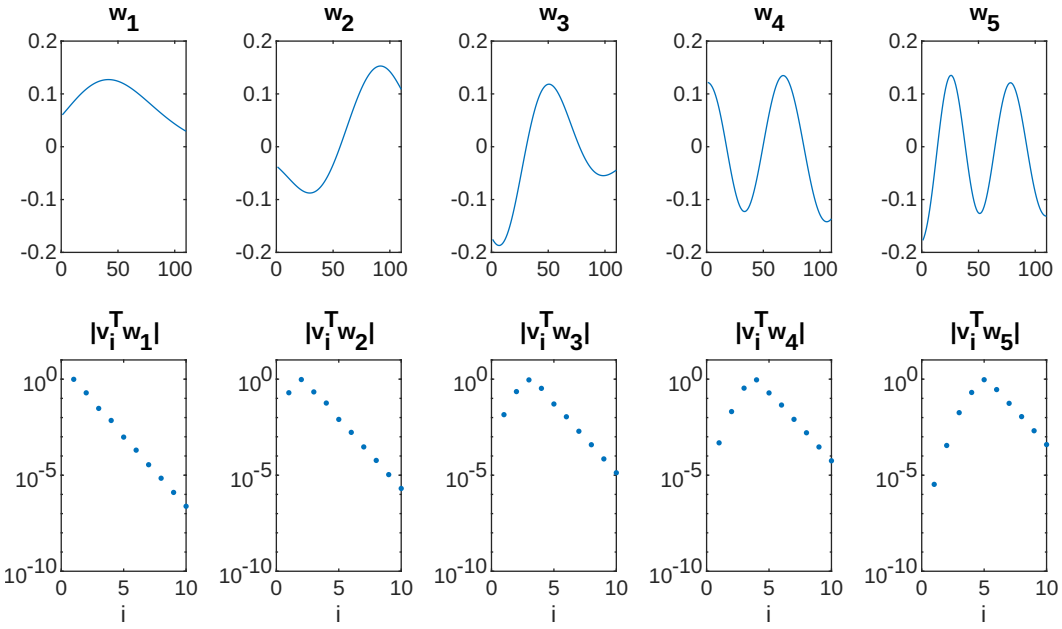
Vzhledem k tomu, že matice A je obecně obdélníková, nemůžeme vždy volit $M = A$ – v Kapitole 2.1 budeme namísto toho používat $M = A^TA$ a $\phi = A^Tb$. Problémem, který budeme potřebovat vyřešit, bude nalezení báze $\mathcal{K}_k(M,\phi)$: posloupnost vektorů $\phi, M\phi, M^2\phi, \ldots$ totiž typicky[2] rychle konverguje k vlastnímu vektoru M příslušnému největšímu vlastnímu číslu M a pokud bychom z těchto vektorů vytvořili sloupce matice D_k, byla by v důsledku mimořádně špatně podmíněná. Tento jev je dobře vidět na Obrázku 1.7. Pro daný Krylovův prostor budeme proto vytvářet ortonormální bázi. V následujících kapitolách nejprve přiblížíme možné metody, jak tuto ortonormální bázi vytvořit, a pro ně následně zanalyzujeme šíření šumu.

Vzhledem k tomu, že přenásobení maticí A (a tím spíše i A^TA) potlačuje vyšší frekvence, budou Krylovovy prostory přirozeně generovány hladkými vektory. Vytvoříme-li navíc ortogonální bázi Krylovova prostoru, budou vektory χ_i této báze blízké singulárním vektorům v_j, přičemž $i \approx j$ [Han10, 6.3.1]. Tento jev je ilustrován na Obrázku 1.8.

[2]Výjimku představuje například situace, kdy dominantnímu vlastnímu číslu matice M přísluší více vlastních vektorů, mezi nimiž může posloupnost oscilovat.



Obrázek 1.7: Ilustrace toho, jak posloupnost vektorů $d_j = (A^T A)^{j-1} A^T b$ konverguje k vlastnímu vektoru příslušnému největšímu vlastnímu číslu. V horním řádku vidíme jednotlivé takto získané vektory d_j (pro přehlednost jsou normalizovány), spodní řádek pak ukazuje skalární součiny $v_i^T d_j$. Vidíme, že postupně ve vektorech d_j převažuje komponenta ve směru vektoru v_1.



Obrázek 1.8: Analogické grafy jako v Obrázku 1.7. Vektory w_j jsou vytvořeny pomocí Arnoldiho algoritmu (detailněji popsaného v Kapitole 3), díky čemuž jsou v nich postupně zastoupeny i vyšší frekvence. Zároveň vidíme, že ve vektorech w_j tvoří dominantní složku právě komponenta ve směru v_j.

2. Golub-Kahanova bidiagonalizace a metoda LSQR

V této kapitole představíme algoritmus Golub-Kahanovy iterační bidiagonalizace a z něj vycházející krylovovskou metodu LSQR a shrneme zde analýzu propagace šumu v této metodě, zpracovanou v článcích [HPS09] a [HKP17]. Výklad bude doplněn o vlastní důkazy a původní numerické experimenty.

Základní ideou metody LSQR, představené poprvé v článku [PS82], je řešení lineárního problému $Ax = b$ s obecně obdélníkovou maticí A pomocí řešení *soustavy normálních rovnic*

$$A^T Ax = A^T b.$$

Aproximace tohoto řešení hledá metoda LSQR iterativně, v k-té iteraci hledá v Krylovově prostoru $\mathcal{K}_k(A^T A,A^T b)$, přičemž pro vytvoření ortonormální báze tohoto prostoru využívá algoritmus *Golub-Kahanovy iterativní bidiagonalizace*. Tento algoritmus, popsaný poprvé v [GK65], nyní podrobně popíšeme.

2.1 Golub-Kahanova iterační bidiagonalizace

Položme na úvod vektory $w_0 = 0$ a $s_1 = b/\|b\|$ a číslo $\beta_1 = \|b\|$. Pro $j = 1,2,\ldots$ budeme nyní konstruovat dvě ortogonální posloupnosti vektorů s_j a w_j takto:

$$\begin{aligned} \alpha_j w_j &= A^T s_j - \beta_j w_{j-1}, \|w_j\| = 1, \alpha_j \geq 0, \\ \beta_{j+1} s_{j+1} &= A w_j - \alpha_j s_j, \|s_{j+1}\| = 1, \beta_{j+1} \geq 0. \end{aligned} \tag{2.1}$$

a to do té doby, než $\alpha_j = 0, \beta_{j+1} = 0$, nebo než $j = n$ – v praxi budeme chtít proces pochopitelně zastavit dříve a omezit se jen na nějaký menší Krylovův podprostor. Pro jednoduchost předpokládejme, že bidiagonalizace neskončí dříve, než v iteraci $j + 1$, tj. že $\alpha_\iota \neq 0 \neq \beta_{\iota+1}$ pro $\iota \leq j$.

Věta 5. *Posloupnost vektorů $\{s_i\}_{i=1}^j$ generovaná Golub-Kahanovou bidiagonalizací tvoří ortonormální bázi Krylovova prostoru $\mathcal{K}_j(AA^T,b)$ a posloupnost vektorů $\{w_i\}_{i=1}^j$ generovaná Golub-Kahanovou bidiagonalizací ortonormální bázi Krylovova prostoru $\mathcal{K}_j(A^T A,A^T b)$.*

Důkaz. Tvrzení dokážeme indukcí. Pro $j = 1$ obě vlastnosti platí, $s_1 \in \text{span}\{b\}$ a $w_1 = A^T s_1 \in \text{span}\{A^T b\}$.

Předpokládejme nyní, že tvrzení platí pro $j = \tau$. Ukážeme nejprve, že vektor $s_{\tau+1} \in \mathcal{K}_{\tau+1}(AA^T,b)$. Vektor $s_{\tau+1}$ splňuje $\beta_{\tau+1} s_{\tau+1} = A w_\tau - \alpha_\tau s_\tau$,

$$A w_\tau \in A\mathcal{K}_\tau(A^T A,A^T b) \subseteq \mathcal{K}_{\tau+1}(AA^T,b)$$

a podle předpokladu $\alpha_\tau s_\tau \in \mathcal{K}_\tau(AA^T,b) \subseteq \mathcal{K}_{\tau+1}(AA^T,b)$, proto také vektor $s_{\tau+1} \in \mathcal{K}_{\tau+1}(AA^T,b)$. Nyní ukážeme, že $w_{\tau+1} \in \mathcal{K}_{\tau+1}(A^T A,A^T b)$. Tento vektor splňuje $\alpha_{\tau+1} w_{\tau+1} = A^T s_{\tau+1} - \beta_{\tau+1} w_\tau$,

$$A^T s_{\tau+1} \in A^T \mathcal{K}_\tau(AA^T,b) \subseteq \mathcal{K}_{\tau+1}(A^A,A^T b)$$

a podle předpokladu $\beta_{\tau+1} w_\tau \in \mathcal{K}_\tau(A^T A,A^T b)$, proto $w_{\tau+1} \in \mathcal{K}_{\tau+1}(A^A A,A^T b)$.

Zbývá nám ověřit ortogonalitu: pro součin $s_{\tau+1}^T s_\iota$, kde $\iota < \tau$, platí

$$\beta_{\tau+1} s_{\tau+1}^T s_\iota = w_\tau^T A^T s_\iota - \alpha_\tau s_\tau^T s_\iota = w_\tau^T (\alpha_\iota w_\iota + \beta_\iota w_{\iota-1}) - 0 = 0$$

z ortogonality posloupností $\{s_j\}$ a $\{w_j\}$. Pro součin $w_{\tau+1}^T w_\iota$ analogicky platí

$$\alpha_{\tau+1} w_{\tau+1}^T w_\iota = s_\tau^T A^{T^T} w_\iota - \beta_\tau w_{\tau-1}^T w_\iota = s_\tau^T (\beta_{\iota+1} s_{\iota+1} + \alpha_\iota s_\iota) - 0 = 0.$$

□

Celý tento proces můžeme zapsat maticově – označme $S_j = [s_1, \ldots, s_j] \in \mathbb{R}^{m \times j}$ a $W_j = [w_1, \ldots, w_j] \in \mathbb{R}^{n \times j}$, a dále definujme matice

$$L_j = \begin{pmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \ddots & \ddots & \\ & & \beta_j & \alpha_j \end{pmatrix} \in \mathbb{R}^{j \times j}, L_{j+} = \begin{pmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \ddots & \ddots & \\ & & \beta_j & \alpha_j \\ & & & \beta_{j+1} \end{pmatrix} \in \mathbb{R}^{(j+1) \times j},$$

Celý výpočet (2.1) můžeme nyní zapsat jako

$$\begin{aligned} A^T S_j &= W_j L_j^T \\ A W_j &= S_{j+1} L_{j+}. \end{aligned} \tag{2.2}$$

Vztahy, které jsme zde odvodili, dosadíme v Sekci 2.3 do rovnice (1.8) a popíšeme zcela praktickou metodu řešení lineárního problému $Ax = b$.

2.2 Šíření šumu v Golub-Kahanově bidiagonalizaci

V předchozí sekci jsme popsali algoritmus Golub-Kahanovy bidiagonalizace a ukázali jsme, že v přesné aritmetice a bez šumu bude korektně fungovat. Jak ale víme z Kapitoly 1, při studiu inverzních úloh pracujeme s pravou stranou b zanesenou šumem. V této sekci popíšeme, jak se tento šum propaguje do bázových vektorů, které Golub-Kahanova bidiagonalizace produkuje. Pro zbytek této kapitoly budeme pracovat v přesné aritmetice, zanedbáme proto zaokrouhlovací chyby a z nich plynoucí ztrátu ortogonality, jediným zdrojem šumu pro nás tedy bude pravá strana b.

První rovnici vztahu (2.2) můžeme nyní přenásobit maticí A. Dostaneme rovnost

$$A A^T S_j = A W_j L_j^T = S_{j+1} L_{j+} L_j^T. \tag{2.3}$$

Za povšimnutí stojí matice

$$L_{j+} L_j^T = \begin{pmatrix} \alpha_1^2 & \alpha_1\beta_2 & & \\ \alpha_1\beta_2 & \alpha_2^2 + \beta_2^2 & \ddots & \\ & \ddots & \ddots & \alpha_{j-1}\beta_j \\ & & \alpha_{j-1}\beta_j & \alpha_j^2 + \beta_j^2 \\ & & & \alpha_j \beta_{j+1} \end{pmatrix}.$$

Rovnici (2.3) díky ní můžeme přepsat jako

$$AA^T S_j = S_j L_j L_j^T + \alpha_j \beta_{j+1} s_{j+1} e_j^T,$$

díky čemuž dokážeme vektor s_{j+1} vyjádřit rekurzivně jako

$$\begin{aligned} \alpha_j \beta_{j+1} s_{j+1} &= AA^T s_j - (\alpha_j^2 + \beta_j^2) s_j - \alpha_{j-1} \beta_j s_{j-1}, \\ \alpha_1 \beta_2 s_2 &= AA^T s_1 - \alpha_1^2 s_1 = \frac{1}{\beta_1}(AA^T - \alpha_1^2 I) b. \end{aligned} \tag{2.4}$$

Z toho je ihned vidět následující poznatek.

Lemma 6. *Vektor s_{j+1} vytvořený Golub-Kahanovou bidiagonalizací lze vyjádřit jako*

$$s_{j+1} = \varphi_j(AA^T) b, \tag{2.5}$$

kde $\varphi_j \in \mathcal{P}_j$ je polynom splňující rekurenci

$$\begin{aligned} \alpha_j \beta_{j+1} \varphi_{j+1} &= AA^T \varphi_j - (\alpha_j^2 + \beta_j^2) \varphi_j - \alpha_{j-1} \beta_j \varphi_{j-1}, \\ \alpha_1 \beta_2 \varphi_2 &== \frac{1}{\beta_1}(AA^T - \alpha_1^2 I). \end{aligned}$$

Do rovnice (2.5) můžeme nyní ze vztahu $Ax - e = b$ dosadit – dostaneme

$$s_{j+1} = \varphi_j(AA^T)(Ax - e) = \varphi_j(AA^T)Ax + [\varphi_j(AA^T) - \varphi_j(0)]e + \varphi_j(0)e,$$

kde všechny členy polynomu $\varphi_j(AA^T) - \varphi_j(0)$ obsahují násobek matice AA^T. Vektor s_{j+1} můžeme díky tomu rozdělit na dva sčítance: na vektor

$$s_{j+1}^{LF} = \varphi_j(AA^T)Ax + [\varphi_j(AA^T) - \varphi_j(0)]e,$$

jehož všechny složky obsahují násobek matice AA^T a pro dostatečně malé j jsou tedy hladké, a na vektor $\varphi_j(0)e$, tj. vektor šumu přenásobený absolutním členem φ_j.

Vidíme, že pro odhad velikosti šumu se nám hodí popsat co nejdetailněji $\varphi_j(0)$. Drobnou modifikací vztahu (2.4) můžeme pro $\varphi_j(0)$ vytvořit rekurentní vztah:

$$\begin{aligned} \alpha_j \beta_{j+1} \varphi_j(0) &= -(\alpha_j^2 + \beta_j^2)\varphi_{j-1}(0) - \alpha_{j-1}\beta_j \varphi_{j-2}(0), \\ \alpha_1 \beta_2 \varphi_1(0) &= -\alpha_1^2 \varphi_0(0), \end{aligned}$$

přičemž $\varphi_0(0) = 1/\beta_1$. Koeficient $\varphi_1(0)$ můžeme snadno vyjádřit:

$$\begin{aligned} \alpha_1 \beta_2 \varphi_1(0) &= \frac{-\alpha_1^2}{\beta_1}, \\ \varphi_1(0) &= -\frac{\alpha_1}{\beta_1 \beta_2}. \end{aligned}$$

Pro obecné j popisuje $\varphi_j(0)$ následující věta.

Věta 7. *Absolutní člen polynomu φ_j získaného Golub-Kahanovou bidiagonalizací je roven $\varphi_j(0) = (-1)^j \dfrac{1}{\beta_{j+1}} \displaystyle\prod_{i=1}^{j} \frac{\alpha_i}{\beta_i}$.*

Důkaz. Matematickou indukcí dokážeme rekurentní vztah $\varphi_i(0) = -\frac{\alpha_i}{\beta_{i+1}}\varphi_{i-1}(0)$ pro $i = 1, \ldots, j$. Pro $i = 1$ tvrzení zjevně platí, předpokládejme nyní, že platí pro přirozené $i - 1 < j$. Pak

$$\begin{aligned}
\alpha_i\beta_{i+1}\varphi_i(0) &= -(\alpha_i^2 + \beta_i^2)\varphi_{i-1}(0) - \alpha_{i-1}\beta_i\varphi_{i-2}(0),\\
\alpha_i\beta_{i+1}\varphi_i(0) &= -(\alpha_i^2 + \beta_i^2)\Big(-\frac{\alpha_{i-1}}{\beta_i}\Big)\varphi_{i-2}(0) - \alpha_{i-1}\beta_i\varphi_{i-2}(0),\\
\alpha_i\beta_{i+1}\varphi_i(0) &= \frac{\alpha_i^2\alpha_{i-1}}{\beta_i}\varphi_{i-2}(0) + \alpha_{i-1}\beta_i\varphi_{i-2}(0) - \alpha_{i-1}\beta_i\varphi_{i-2}(0),\\
\alpha_i\beta_{i+1}\varphi_i(0) &= \frac{\alpha_i^2\alpha_{i-1}}{\beta_i}\varphi_{i-2}(0),\\
\varphi_i(0) &= \frac{\alpha_i\alpha_{i-1}}{\beta_{i+1}\beta_i}\varphi_{i-2}(0) = -\frac{\alpha_i}{\beta_{i+1}}\varphi_{i-1}(0).
\end{aligned}$$

□

Vidíme tedy, že zanesení vektoru s_j šumem je přímo závislé na hodnotě $\varphi_j(0)$, pro nějž již máme explicitní vyjádření. Otázkou nyní je, jak se bude $\varphi_j(0)$ chovat pro rostoucí j – tuto otázku blíže studuje [HPS09]. V Kapitole 1.4 jsme pojednávali o úrovni κ, od které jsou součiny $u_i^T b$ tvořeny především šumem, a podobné chování budeme pozorovat i zde. Bez přítomnosti šumu jsou vektory u_i a s_j jsou spjaty stejně jako vektory v_i a w_j, jejichž spojitost ukazuje Obrázek 1.8 – s_j má dominantní složky odpovídající vektoru u_j.

Vstoupí-li do hry šum, platí toto provázání pouze do určité chvíle – pro určité $j = j_{rev}$ ([HPS09] jej nazývají *noise-revealing iteration*) dosáhne $\varphi_{j_{rev}}(0)$ takové úrovně, že ve vektoru $s_{j_{rev}+1}$ převládne šum. To ilustruje Obrázek 2.1. Vzhledem k tomu, že bílý šum obsahuje všechny frekvence ve stejné míře, budou ve vektoru $s_{j_{rev}+1}$ stejně zastoupeny složky vektorů u_ι pro $\iota \geq j_{rev}+1$. Iteraci j_{rev} lze poznat na první pohled, což však pro implementaci nestačí – možné metody její strojové detekce shrnuje kromě [HPS09] i například [Vas11] a [Mic13].

[HPS09] dále ukazují, že pro $j < j_{rev}$ bude $\varphi_j(0)$ v průměru iteraci od iterace růst – dosadíme-li do rovnice (2.1) singulární rozklad $A = U\Sigma V^T$, dostaneme pro první iteraci

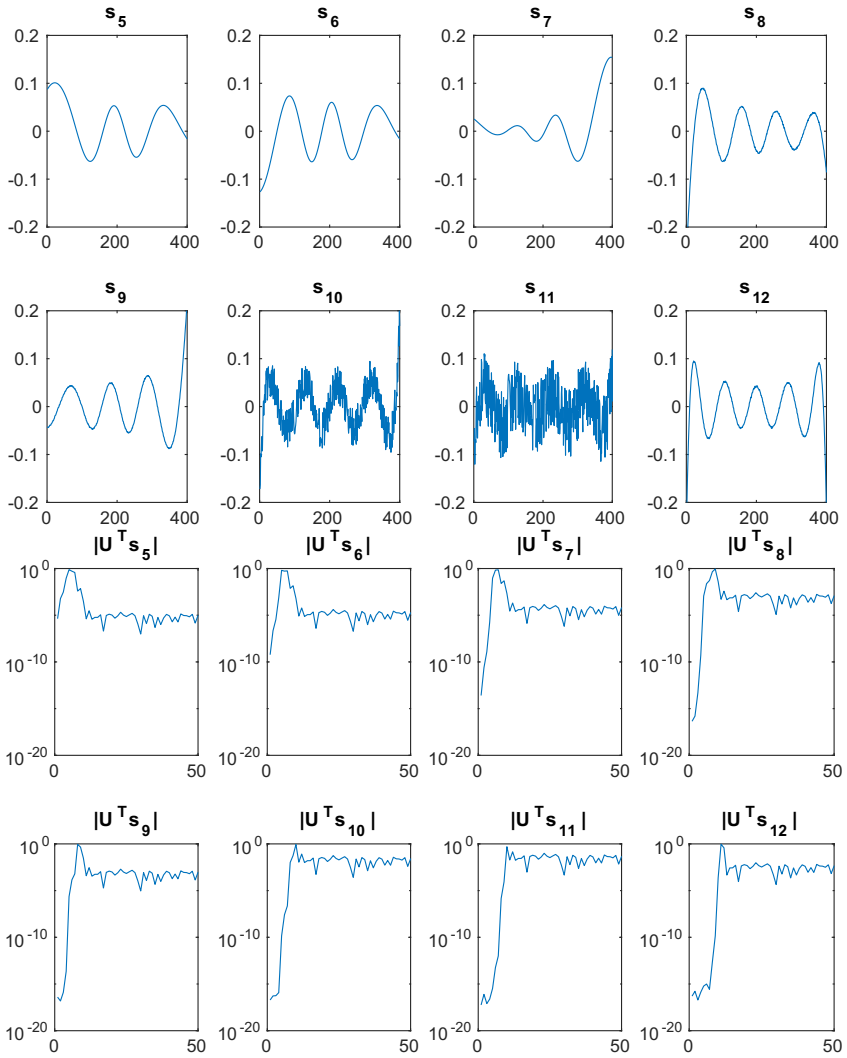
$$\alpha_1 V^T w_1 = \Sigma U^T s_1, \beta_2 U^T s_2 = \Sigma V^T w_1 - \alpha_1 U^T s_1,$$

z čehož vidíme, že zatímco $V^T w_1$ a $U^T s_1$ bude obsahovat stejné spektrální komponenty, $U^T s_2$ je oproti $U^T s_1$ ortogonalizováno, $U^T s_1$ a $V^T w_1$ se tedy musí vyrušit, a proto β_2 musí být znatelně menší než α_1. Stejný princip platí i pro obecné j:

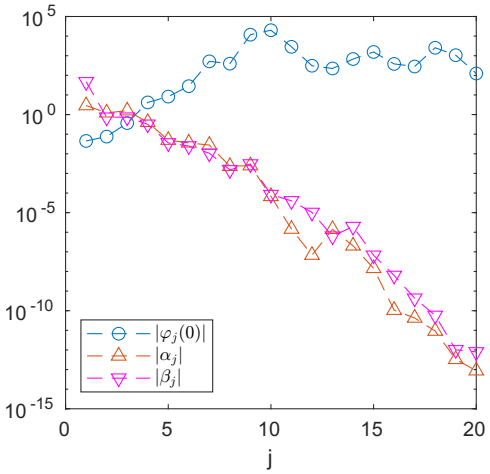
$$\alpha_j V^T w_j = \Sigma U^T s_j - \beta_j V^T w_{j-1}, \beta_{j+1} U^T s_{j+1} = \Sigma V^T w_j - \alpha_j U^T s_j.$$

V $U^T s_j$ budou převažovat jiné komponenty než v $V^T w_{j-1}$ a tedy α_j a β_j mohou být řádově stejné. Naopak v $U^T s_j$ a $V^T w_j$ převažuje tatáž komponenta, která se v zájmu ortogonality $U^T s_{j+1}$ a $U^T s_j$ musí opět vyrušit – tudíž $|\beta_{j+1}|$ musí být mnohem nižší než $|\alpha_j|$. Tento pokles ilustruje Obrázek 2.2.

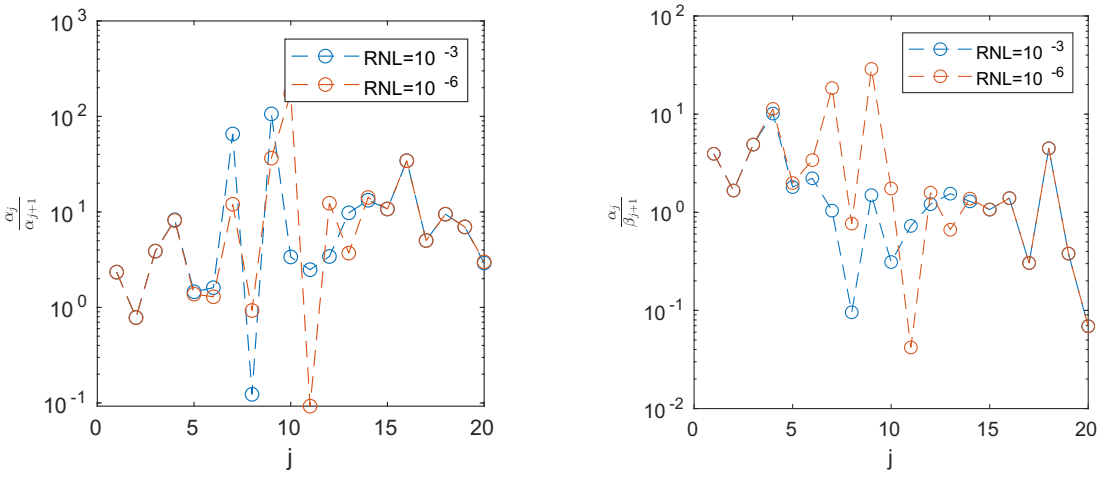
Úvaha z předchozích odstavců platí pouze do dosažení j_{rev}. Zaměřme se nyní na chování α_j a β_j ve chvíli dosažení j_{rev}. V této iteraci je vektor $s_{j_{rev}+1}$ zcela zanesen šumem, v důsledku toho v něm žádná komponenta nepřevažuje a je



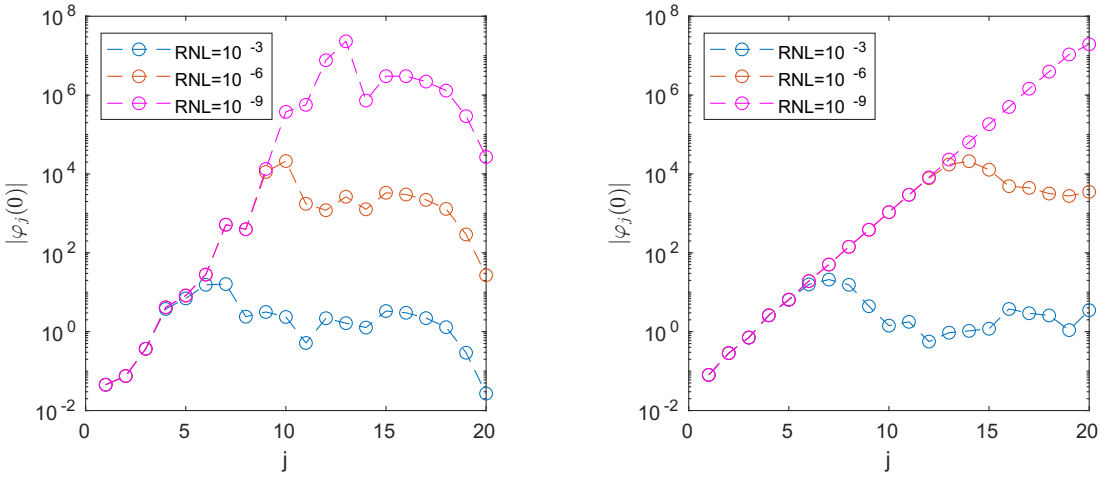
Obrázek 2.1: Jednotlivé vektory s_j a velikosti prvních 50 složek jejich spektra, tj. prvních 50 složek vektoru $|U^T s_j|$ pro modelovou úlohu `shaw(400)` *s šumem o $RNL = 10^{-6}$. Vektor s_{11} je masivně zanesen šumem, vidíme tedy, že $j_{rev} = 10$. Tomu odpovídá i jeho spektrum, v němž jsou všechny vektory u_ι pro $\iota > 10$ zastoupeny rovnoměrně.*



Obrázek 2.2: Koeficienty α_j, β_j a $|\varphi_j(0)|$ pro úlohu `shaw(400)` *se šumem o $RNL = 10^{-6}$.*



Obrázek 2.3: Pokles koeficientu α_j a podílu α_j/β_{j+1} pro dvě různé úrovně šumu v úloze `shaw(400)`*. Pro $RNL = 10^{-6}$ jsme již na Obrázku 2.1 viděli, že $j_{rev} = 10$, pro $RNL = 10^{-3}$ je touto iterací nejspíše $j_{rev} = 7$.*



(a) `shaw(400)` *(b)* `gravity(400)`

Obrázek 2.4: $|\varphi_j(0)|$ pro dvě modelové úlohy. Na průběhu pro `shaw(400)` *s $RNL = 10^{-6}$ vidíme, že maxima dosahuje pro $j = 10$, což je právě j_{rev} z Obrázku 2.1.*

v něm tedy v nezanedbatelné míře přítomna i stejná spektrální komponenta jako v $V^T w_{j_{rev}}$. Tento jev ilustruje Obrázek 2.1. Aby tím pádem mohlo být $V^T w_{j_{rev}}$ ortogonální na $V^T w_{j_{rev}+1}$, musí se oba vektory výrazně vyrušit, a proto $|\alpha_{j_{rev}+1}|$ musí být výrazně menší než $|\beta_{j_{rev}+1}|$. V tuto chvíli proto $|\alpha_j|$ mimořádně poklesne. Podíly $|\alpha_j|/|\beta_{j+1}|$ jsou proto počínaje touto iterací výrazně menší, $|\varphi_j(0)|$ tím pádem při dosažení j_{rev} rovněž klesne a v dalších iteracích již neroste zdaleka tak rychle. Tento pokles $|\alpha_{j_{rev}+1|}$ ilustruje Obrázek 2.3.

Obrázek 2.4 pak ukazuje chování $|\varphi_j(0)|$ ve dvou různých modelových úlohách pro tři různé hladiny šumu. Nepřekvapivě vidíme, že čím nižší RNL, tím vyšší musí být $|\varphi_j(0)|$, aby k projevení šumu došlo.

Analogickým způsobem jako v Lemmatu 6 můžeme vyjádřit i vektor w_j. Druhou rovnici vztahu (2.2) přenásobíme maticí A^T, což nám dá rovnost

$$A^T A W_j = A^T S_{j+1} L_{j+} = W_{j+1} L_{j+1}^T L_{j+}. \tag{2.6}$$

V této rovnici figuruje matice

$$L_{j+1}^T L_{j+} = \begin{pmatrix} \alpha_1^2+\beta_2^2 & \alpha_2\beta_2 & & \\ \alpha_2\beta_2 & \alpha_2^2+\beta_3^2 & \ddots & \\ & \ddots & \ddots & \alpha_j\beta_j \\ & & \alpha_j\beta_j & \alpha_j^2+\beta_{j+1}^2 \\ & & & \alpha_{j+1}\beta_{j+1} \end{pmatrix},$$

díky níž můžeme rovnici (2.6) přepsat jako

$$A^T A W_j = W_j L_j^T L_j + \beta_j^2 w_j e_j^T + \alpha_{j+1}\beta_{j+1} w_{j+1} e_j^T.$$

Vektor w_j proto můžeme analogicky jako v rovnici (2.4) vyjádřit rekurzivně jako

$$\begin{aligned} \alpha_{j+1}\beta_{j+1} w_{j+1} &= A^T A w_j - (\alpha_j^2+\beta_{j+1}^2) w_j - \alpha_j\beta_j w_{j-1}, \\ \alpha_2\beta_2 w_2 &= A^T A w_1 - (\alpha_1^2+\beta_2^2) w_1. \end{aligned}$$

Tato rovnost nám v kombinaci s tím, že

$$w_1 = \frac{A^T b}{\alpha_1\beta_1},$$

dává následující důsledek:

Lemma 8. *Vektor w_{j+1} generovaný Golub-Kahanovou bidiagonalizací můžeme vyjádřit jako*

$$w_{j+1} = \xi_j(A^T A) A^T b,$$

kde $\xi_j \in \mathcal{P}_j$ je polynom splňující rekurenci.

$$\begin{aligned} \alpha_{j+1}\beta_{j+1}\xi_{j+1} &= A^T A \xi_j - (\alpha_j^2+\beta_{j+1}^2)\xi_j - \alpha_j\beta_j\xi_{j-1}, \\ \alpha_2\beta_2\xi_2 &= A^T A A^T - (\alpha_1^2+\beta_2^2) A^T. \end{aligned}$$

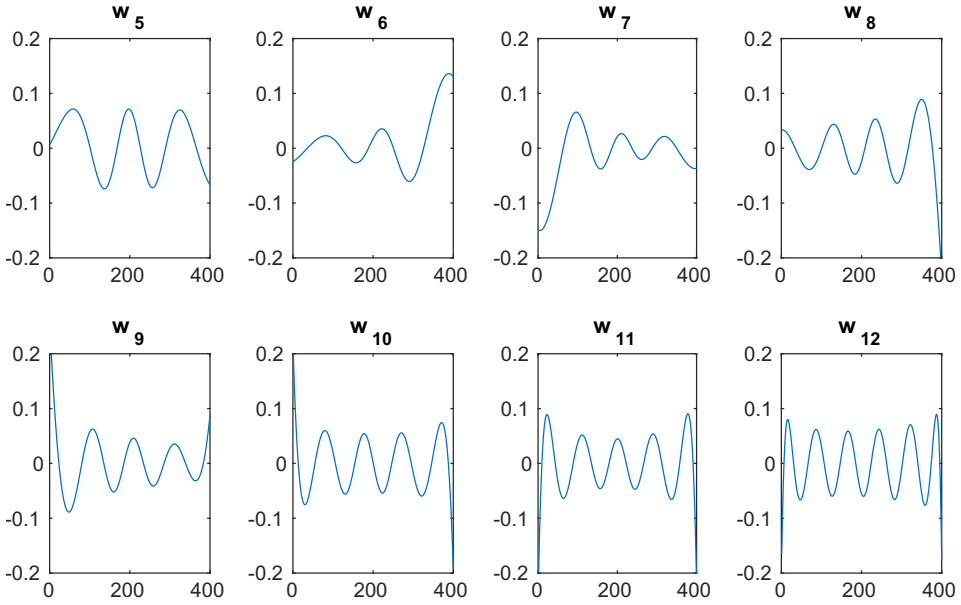
Vidíme, že na rozdíl od sekvence vektorů s_j je posloupnost vektorů w_j hned od prvního vektoru přenásobena maticí A^T, která v ní proto potlačuje vyšší frekvence ze šumu. Jak ilustruje Obrázek 2.5, vektory w_j zůstávají hladké i po dosažení j_{rev}.

2.3 Metoda LSQR a její reziduum

V této sekci popíšeme metodu LSQR [PS82] a s využitím poznatků z předchozí kapitoly ukážeme, jak závisí reziduum této metody na koeficientu $\varphi_j(0)$.

Jak jsme již popsali v úvodu této kapitoly, metoda LSQR řeší obecný problém $Ax = b$ s obdélníkovou maticí A. Pro jednoduchost analýzy budeme i zde předpokládat, že matice A má plnou sloupcovou hodnost. Pokud A není čtvercová, typicky nebude existovat přesné x splňující rovnost $Ax = b$. Namísto toho však můžeme hledat řešení tohoto problému ve smyslu nejmenších čtverců, tj. hledat takové $\hat{x}$, aby $\|A\hat{x} - b\|_2$ bylo co nejmenší. Takové řešení splňuje rovnost [BT18, Věta 8.91]

$$A^T A \hat{x} = A^T b. \tag{2.7}$$



Obrázek 2.5: Jednotlivé vektory w_j pro modelovou úlohu `shaw(400)` *se šumem o $RNL = 10^{-6}$. Oproti vektorům s_j pro stejnou úlohu se stejnou RNL, které jsme viděli na Obrázku 2.1 jsou všechny vektory hladké.*

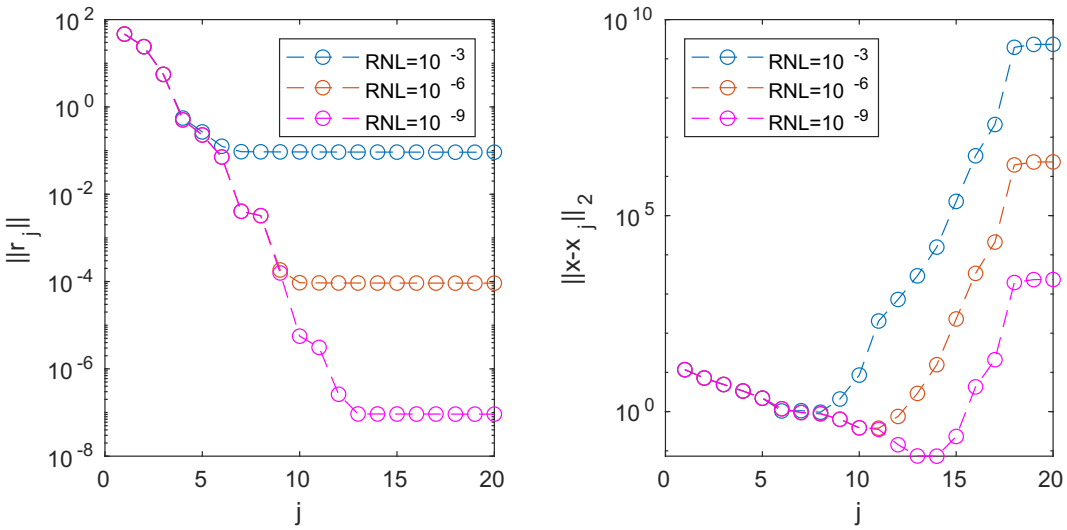
Metoda LSQR funguje iterativně a v j-té iteraci hledá aproximaci $x_j \approx \hat{x}$ takovou, že $x_j \in K_j(A^TA, A^Tb)$. Tuto aproximaci x_j proto najdeme jako $x_j = W_j y_j$, kde

$$y_j = \text{argmin}_y \|AW_j y - b\|_2 = \text{argmin}_y \|S_{j+1} L_{j+} y - \beta_1 S_{j+1} e_1\|_2 = \\ = \text{argmin}_y \|L_{j+} y - \beta_1 e_1\|_2. \quad (2.8)$$

Druhá rovnost je důsledkem (2.2), poslední rovnost platí díky vzájemné ortogonalitě sloupcových vektorů matice S_{j+1}. Dostáváme tedy problém nejmenších čtverců, který nepracuje s původní maticí, ale s bidiagonální maticí L_{j+} a je tedy snadné jej vyřešit pomocí QR rozkladu.

Je důležité si uvědomit, že metoda LSQR sama o sobě nestačí k řešení diskrétní inverzní úlohy – provedeme-li dostatečný počet iterací, LSQR nás dovede k řešení $x_j \approx x^{naive}$, o němž jsme již v Sekci 1.4 ukázali, že je nepoužitelné. Důležitá však je vlastnost metody LSQR, kterou ukazuje Obrázek 2.6: v důsledku rovnosti (2.8) klesá norma $\|x_j - x^{naive}\|_{A^TA}$, proto obvykle do dosažení iterace j_{rev} klesá také norma $\|x_j - x\|$. Po dosažení j_{rev} převáží ve vektorech s_j šum, který následně ovlivní i vektory w_j a norma chyby začne postupně růst. Tomuto procesu říkáme *semikonvergence* metody LSQR. Je tedy klíčové iteraci j_{rev} detekovat a metodu LSQR včas zastavit, různým metodám detekce j_{rev} se věnují např. [HPS09], [Vas11] nebo [Mic13].

Vyjdeme nyní z článku [HKP17] a vyjádříme reziduum $r_j^{LSQR} = b - Ax_j$ pro aproximaci x_j vzniklou po j iteracích metody LSQR. Využijeme k tomu koeficient $\varphi_j(0)$, na němž závisí propagace šumu ve vektorech s_j. Již nyní můžeme s určitostí říci, že norma $\|r_j^{LSQR}\|$ je nerostoucí, jelikož metoda LSQR v každé své iteraci reziduum minimalizuje (a množina, na níž jej minimalizuje, je podle (2.8) v každé iteraci větší a větší).



Obrázek 2.6: Norma rezidua a chyby metody LSQR pro úlohu `shaw(400)` *pro tři různé úrovně šumu. Vidíme, že pokles rezidua prudce zpomaluje (a prakticky zastavuje) při dosažení j_{rev} (pro $RNL = 10^{-6}$ jsme již viděli, že $j_{rev} = 10$). Norma chyby začíná krátce po dosažení j_{rev} růst – tento růst však nemusí přijít hned, norma chyby může ještě v další iteraci klesnout.*

Věta 9. *Norma rezidua r_j^{LSQR} je rovna*

$$\|r_j^{LSQR}\| = \frac{1}{\sqrt{\sum_{\iota=0}^{j} \varphi_\iota(0)^2}} \tag{2.9}$$

Důkaz. V důkazu si pomůžeme obecnější teorií toho, jak spolu souvisí normy reziduí v různých krylovovských metodách, kterou popisuje [Saa03, 6]. Konkrétně si vyjádříme reziduum, které dostaneme z metody CRAIG [Cra55], pro niž je analýza rezidua jednodušší než pro LSQR, a z něj pak dovodíme $\|r_j^{LSQR}\|$.

Označme r_j^{CRAIG} reziduum, které bychom dostali, kdyby $x_j^{CRAIG} = W_j y_j^{CRAIG}$ mělo minimalizovat vzdálenost od x^{naive}. V takovém případě bude $r_j^{CRAIG} = c_j s_{j+1}$ pro nějaké c_j, které se mění iteraci od iterace. Zároveň $s_{j+1} = \varphi_j(AA^T)b$ a

$$r_j^{CRAIG} = b - AW_j y_j^{CRAIG} = \beta_1 S_{j+1} e_1 - S_{j+1} L_{j+} y_j^{CRAIG} = \Pi_j(A^T A)b,$$

kde Π_j je polynom stupně j s konstantním členem rovným 1. Zkombinujeme-li tyto informace dohromady, musí $c_j \varphi_j(0) = \Pi_j(0) = 1$ a tedy $c_j = \varphi_j^{-1}(0)$. Podle [Saa03, Věta 6.14] proto bude pro reziduum r_j^{LSQR} platit

$$\|r_j^{LSQR}\| = \frac{1}{\sqrt{\sum_{\iota=0}^{j} (\|r_\iota^{CRAIG}\|)^{-2}}} = \frac{1}{\sqrt{\sum_{\iota=0}^{j} \varphi_\iota(0)^2}}.$$

□

Věta 10. *Reziduum r_j^{LSQR} je rovno*

$$r_j^{LSQR} = \frac{1}{\sum_{\iota=0}^{j} \varphi_\iota(0)^2} \sum_{\iota=0}^{j} \varphi_\iota(0) s_{\iota+1}. \tag{2.10}$$

Důkaz. Označme $p_j = \beta_1 e_1 - L_{j+} y_j$. Podle rovnice (2.8) je

$$y_j = \text{argmin}_y \|L_{j+} y - \beta_1 e_1\|$$

a tedy $L_{k+}^T p_k = 0$. Z toho plyne, že jednotlivé složky vektoru p_k splňují pro $\iota = 1, \ldots, j$

$$\alpha_\iota e_\iota^T p_j + \beta_{\iota+1} e_{\iota+1}^T p_j = 0.$$

Z toho můžeme usuzovat, že

$$p_j = c_j \begin{pmatrix} \varphi_0(0) \\ \varphi_1(0) \\ \vdots \\ \varphi_j(0) \end{pmatrix},$$

kde c_j se iteraci od iterace může měnit. Protože norma

$$\|p_j\| = \|r_j^{LSQR}\| = \frac{1}{\sqrt{\sum_{\iota=0}^{j} \varphi_\iota(0)^2}}, \quad \text{je koeficient} \quad c_j = \frac{1}{\sum_{\iota=0}^{j} \varphi_\iota(0)^2}.$$

Dosazením do rovnosti $r_j^{LSQR} = S_{j+1} p_j$ dostaneme požadovanou rovnost (2.10). □

Věta 10 jinými slovy říká, že míra zastoupení vektorů s_j v reziduu je tím větší, čím více je v těchto vektorech přítomen šum. Zároveň z Věty 9 vyplývá, že norma rezidua je nerostoucí funkce, která relativně výrazně klesá před dosažením j_{res}, kdy $\varphi_j(0)$ s každou iterací výrazně stoupá. Naopak po dosažení j_{rev} se pokles normy rezidua zastaví, jelikož všechny sčítance $\varphi_\iota(0)$ pro $\iota > j_{rev}$ ve jmenovateli zlomku z (2.9) jsou řádově menší než již dosažený součet. Tento proces ilustruje první graf Obrázku 2.6.

3. Metody založené na Arnoldiho a Lanczosově algoritmu

3.1 Arnoldiho a Lanczosův algoritmus

V této kapitole budeme předpokládat, že matice A je čtvercová regulární – tím pádem můžeme v definici 5 položit přímo $M = A$. Přiblížíme postupně dva algoritmy tvořící ortonormální bázi Krylovova prostoru $\mathcal{K}_j(A,b)$ – Arnoldiho a Lanczosův – a provedeme analýzu propagace šumu v těchto algoritmech analogicky k předchozí kapitole. Zatímco analýza Golub-Kahanovy bidiagonalizace byla spíše kompilací již existujících článků, analýza v této kapitole bude původním dílem autora. Nejprve popíšeme oba algoritmy pro tvorbu ortonormální báze Krylovova prostoru, přičemž jejich formulaci převezmeme z [DTHPS12, 7.1].

Arnoldiho algoritmus

Popíšeme nejprve *Arnoldiho algoritmus*, který pracuje s obecnou čtvercovou regulární maticí A. Algoritmus vychází z Gram-Schmidtovy ortogonalizace posloupnosti $\{b, Ab, A^2b, \dots,\}$. Aby však tyto vektory nekonvergovaly k vlastním vektorům příslušným dominantním vlastním číslům, postupuje se iterativně: pro $j = 1,2,\dots$ postupně vytvoříme ortonormální množinu vektorů $\{w_1,w_2,\dots\}$, kde $w_1 = b/\|b\|$ a $w_{j+1} = z/\|z\|$ pro

$$z = Aw_j - \sum_{i=1}^{j} h_{i,j}w_i, \quad h_{i,j} = w_i^T Aw_j, \quad h_{j+1,j} = \|z\|. \tag{3.1}$$

Vidíme, že tento výpočet pracuje s dlouhými rekurencemi – v každé iteraci ortogonalizujeme vektor Aw_j proti všem předchozím vektorům $w_1, w_2, \dots, w_j$. Algoritmus lze prakticky implementovat různými způsoby v závislosti na tom, z jaké varianty Gram-Schmidtova algoritmu vyjdeme. Proces můžeme zapsat i maticově:

$$AW_j = W_{j+1}H_{j+}, \tag{3.2}$$

kde $W_k = [w_1, \dots, w_k]$ a $H_{j+} = (h_{i,k}), 1 \le i \le j+1, 1 \le k \le j$. Prvky matice H_j splňují $h_{i,k} = 0$ pro $i > k+1$ – takovouto matici nazýváme *horní Hessenbergova matice*.

Tuto rovnost můžeme zapsat také jako

$$AW_j = W_jH_j + h_{j+1,j}w_{j+1}e_j^T, \tag{3.3}$$

kde H_j je matice H_{j+} bez posledního řádku. Přenásobíme-li tento vztah zprava maticí W_j^T, dostaneme

$$W_j^T AW_j = H_j.$$

Pokud neřekneme jinak, budeme v celé této kapitole pro jednoduchost předpokládat $k \le st_b(A)$ – dosažení $k = st_b(A)$ by v Arnoldiho algoritmu znamenalo, že $h_{k+1,k} = 0$ a již nemůžeme spočítat další vektor w_{k+1}. V takovém případě se rovnost (3.2) mění na $AW_k = W_kH_k$.

Věta 11. *Posloupnost vektorů* $\{w_i\}_{i=1}^{j}$ *generovaná Arnoldiho algoritmem tvoří ortonormální bázi Krylovova prostoru* $\mathcal{K}_j(A,b)$.

Důkaz. Tvrzení dokážeme matematickou indukcí – pro vektor $w_1 = b/\|b\|$ zjevně platí. Pokud vektory $w_1, \ldots, w_k$ tvoří ortonormální bázi prostoru $\mathcal{K}_k(A,b)$, pak $\sum_{\iota=1}^{k} h_{\iota,k} w_k \in \mathcal{K}_k(A,b)$ a $Aw_k \in A\mathcal{K}_k(A,b)$, proto $w_{k+1} \in \mathcal{K}_{k+1}(A,b)$. Zároveň $w_{k+1}^T w_\iota = 0$ pro $\iota < k$, protože

$$h_{k+1,k} w_{k+1}^T w_\iota = (Aw_k)^T w_\iota - \sum_{i=1}^{k} h_{i,k} w_i^T w_\iota = h_{\iota,k} - h_{\iota,k} = 0.$$

Tím pádem $\{w_i\}_{i=1}^{k+1}$ také tvoří ortonormální bází $\mathcal{K}_{k+1}(A,b)$. □

Lanczosův algoritmus

Předpokládejme pro zbytek této sekce, že matice A je regulární a symetrická[1]. Pro takové zadání budeme výše popsaný algoritmus nazývat *Lanczosův algoritmus* – poprvé jej (v kontextu řešení problému vlastních čísel) popsal [Lan50], podrobněji se jeho numerickému chování věnuje [MS06]. Pro matici H_j v takovém případě platí

$$H_j = W_j^T A W_j = W_j^T A^T W_j = H_j^T.$$

Matice H_j je tedy také symetrická, zároveň však i horní Hessenbergova – tím pádem musí mít nulové všechny prvky vyjma hlavní diagonály a dvou vedlejších diagonál. Je tedy *tridiagonální* a v dalším textu ji tím pádem budeme značit

$$T_j = \begin{pmatrix} \alpha_1 & \beta_2 & & & \\ \beta_2 & \alpha_2 & \beta_3 & & \\ & \beta_3 & \ddots & \ddots & \\ & & \ddots & \ddots & \beta_j \\ & & & \beta_j & \alpha_j \end{pmatrix},$$

kde $\beta_2, \beta_3, \ldots, \beta_j > 0$. Obecně se reálná tridiagonální symetrická matice s kladnými prvky na vedlejších diagonálách nazývá *Jacobiho matice*. Vlastní čísla této matice nazýváme *Ritzova čísla* a jak ukázal [Lan50], v přesné aritmetice Ritzova čísla aproximují vlastní čísla matice A.

Rovnici (3.3) můžeme v takovém případě zapsat jako

$$AW_j = W_j T_j + \beta_{j+1} w_{j+1} e_j^T \tag{3.4}$$

a rekurenci (3.1) jako

$$\beta_{j+1} w_{j+1} = Aw_j - \alpha_j w_j - \beta_j w_{j-1}, \quad \alpha_j = w_j^T A w_j, \quad \|w_{j+1}\| = 1, \tag{3.5}$$

kde $w_1 = b/\|b\|$ a $\beta_2 w_2 = Aw_1 - \alpha_1 w_1$.

Oproti Arnoldiho algoritmu dokážeme pomocí Lanczosova algoritmu vytvořit ortogonální bázi $\mathcal{K}_j(A,b)$ jen za pomoci tříčlenné rekurence, každý z vektorů Aw_j totiž ortogonalizujeme jen proti dvěma předchozím vektorům w_j a w_{j-1}.

[1] V celé práci pracujeme jen s reálnými maticemi, dále popsaný Lanczosův algoritmus však funguje i pro komplexní hermitovské matice.

3.2 Šíření šumu v Lanczosově algoritmu

Pro Lanczosův algoritmus nyní popíšeme šíření šumu analogicky, jako v Sekci 2.2 pro Golub-Kahanovu bidiagonalizaci. Stejně jako v Kapitole 2 budeme v celé této kapitole pracovat v přesné aritmetice, bez zaokrouhlovacích chyb, jediným zdrojem šumu proto bude pravá strana b. Z rekurence (3.5) vyplývá následující vztah, analogický k Lemmatu 6:

Lemma 12. *Vektor w_{j+1} generovaný Lanczosovým algoritmem můžeme vyjádřit jako*

$$w_{j+1} = \varphi_j(A)b, \tag{3.6}$$

kde $\varphi_j \in \mathcal{P}_j$ je polynom stupně j, který splňuje rekurenci

$$\begin{aligned} \beta_{j+1}\varphi_j(A) &= (A - \alpha_j I)\varphi_{j-1}(A) - \beta_j\varphi_{j-2}(A), \\ \beta_2\varphi_1(A) &= (A - \alpha_1 I)\varphi_0(A), \\ \varphi_0(A) &\equiv \|b\|^{-1}. \end{aligned} \tag{3.7}$$

Do rovnosti (3.6) nyní můžeme dosadit $b = Ax - e$. Dostaneme vztah

$$w_{j+1} = \varphi_j(A)(Ax - e) = \varphi_j(A)Ax + [\varphi_j(A) - \varphi_j(0)]e + \varphi_j(0)e.$$

Všechny členy polynomu $\varphi_j(A) - \varphi_j(0)$ jsou přenásobené maticí A, z lemmatu proto vyplývá následující klíčový důsledek.

Lemma 13. *Vektor w_{j+1} generovaný Lanczosovým algoritmem můžeme rozložit na součet*

$$w_{j+1} = w_{j+1}^{LF} + \varphi_j(0)e,$$

kde

$$w_{j+1}^{LF} = \varphi_j(A)(Ax - e) = \varphi_j(A)Ax + [\varphi_j(A) - \varphi_j(0)]e.$$

Toto lemma má fundamentální důsledky: celý vektor w_{j+1}^{LF} obsahuje násobky matice A, a jsou v něm proto potlačeny vysoké frekvence[2], které do něj vnáší šum. Naopak $\varphi_j(0)e$ obsahuje všechny frekvence ze šumu. I v tomto případě bude klíčové nalézt explicitní vyjádření absolutního členu polynomu $\varphi_j(0)$, který tento šum ve vektorech w_j posiluje.

Dosazením nuly do rekurence (3.7) získáme rovnost

$$\begin{aligned} \beta_{j+1}\varphi_j(0) &= -\alpha_j\varphi_{j-1}(0) - \beta_j\varphi_{j-2}(0), \\ \beta_2\varphi_1(0) &= -\alpha_1\varphi_0(0), \\ \varphi_0(0) &= \|b\|^{-1}. \end{aligned} \tag{3.8}$$

Zatímco v Golub-Kahanově bidiagonalizaci bylo vyjádření φ_j veskrze přímočaré (jedná se o jeden jednoduchý součin), rekurence (3.8) vede ke složitějšímu vyjádření. Abychom jej mohli popsat, definujeme nejprve pomocnou množinu $\mathcal{D}_j$.

Definice 6. *Množinu všech posloupností $\{\Delta_\iota\}_{\iota=1}^{\mu}$ tvořených čísly 1 a 2, které mají součet j, označíme $\mathcal{D}_j$. Pro $j = 0$ položme $\mathcal{D}_j = \emptyset$*

Pro každou posloupnost $\{\Delta_\iota\} \in \mathcal{D}_j$ dále definujeme posloupnost částečných součtů $S(\Delta_\iota)$, tj.

$$S(\Delta_\iota) = \sum_{i=1}^{\iota} \Delta_i.$$

[2]Proto jej značíme horním indexem LF – jde o *low frequency component*.

Lze dokázat, že $|\mathcal{D}_j| = F_j$, kde F_j je j-tý člen Fibonacciho posloupnosti[3].

Věta 14. *Absolutní člen polynomu $\varphi_j(0)$ definovaného v rovnosti (3.6) je pro $j = 1,2\ldots$ roven*

$$\varphi_j(0) = \sum_{\{\Delta_\iota\}\in\mathcal{D}_j} \left(\prod_{\iota=1}^{\mu} (-1)^j \alpha_{S(\Delta_\iota)}^{(\Delta_\iota \mod 2)} (-\beta_{S(\Delta_\iota)})^{2(\Delta_\iota+1 \mod 2)} \right) \cdot \prod_{i=2}^{j+1} \beta_i^{-1} \cdot \|b\|^{-1}, \tag{3.9}$$

přičemž $\varphi_0(0) = \|b\|^{-1}$.

Řečeno slovy, jednotlivé sčítance odpovídají posloupnostem z $\mathcal{D}_j$, přičemž v každém sčítanci je zastoupena $\alpha_{S(\Delta_\iota)}$ právě tehdy, když $\Delta_\iota = 1$, a $(-\beta_{S(\Delta_\iota)})^2$ právě tehdy, když $\Delta_\iota = 2$.[4]

Důkaz. Tvrzení dokážeme matematickou indukcí. Pro $j = 1$ tvrzení platí, jelikož

$$\varphi_1(0) = \frac{-\alpha_1\varphi_0(0)}{\beta_2} = \frac{-\alpha_1}{\beta_2\|b\|},$$

a čitatel tohoto zlomku je roven právě (3.9), neboť jediná posloupnost z $\mathcal{D}_1$ je právě jednočlenná posloupnost s jediným členem 1. Předpokládejme nyní platnost tohoto tvrzení pro všechny absolutní členy až po $\varphi_{j-1}(0)$ – ukážeme, že v takovém případě platí i pro $\varphi_j(0)$. Pro jednoduchost budeme nyní značit

$$N_j = \sum_{\{\Delta_\iota\}\in\mathcal{D}_j} \left(\prod_{\iota=1}^{\mu} (-1)^j \alpha_{S(\Delta_\iota)}^{(\Delta_\iota \mod 2)} (-\beta_{S(\Delta_\iota)})^{2(\Delta_\iota+1 \mod 2)} \right).$$

Rekurence (3.8) nám dává vztah

$$\beta_{j+1}\varphi_j(0) = -\alpha_j \frac{N_{j-1}}{\prod_{i=2}^{j}\beta_i\|b\|} - \beta_j \frac{N_{j-2}}{\prod_{i=2}^{j-1}\beta_i\|b\|} = \frac{-\alpha_j N_{j-1} - \beta_j^2 N_{j-2}}{\prod_{i=2}^{j}\beta_i\|b\|},$$

z nějž vidíme rovnost

$$\varphi_j(0) = \frac{-\alpha_j N_{j-1} - \beta_j^2 N_{j-2}}{\prod_{i=2}^{j+1}\beta_i\|b\|}.$$

Zbývá ukázat, že $-\alpha_j N_{j-1} - \beta_j^2 N_{j-2} = N_j$. Z definice N_j vyplývá vztah

$$-\alpha_j N_{j-1} = \sum_{\{\Delta_\iota\}\in\mathcal{D}_{j-1}} \left(\prod_{\iota=1}^{\mu} (-1)^j \alpha_{S(\Delta_\iota)}^{(\Delta_\iota \mod 2)} (-\beta_{S(\Delta_\iota)})^{2(\Delta_\iota+1 \mod 2)} \alpha_j \right),$$

$$-\beta_j N_{j-2} = \sum_{\{\Delta_\iota\}\in\mathcal{D}_{j-2}} \left(\prod_{\iota=1}^{\mu} (-1)^{j-2} \alpha_{S(\Delta_\iota)}^{(\Delta_\iota \mod 2)} (-\beta_{S(\Delta_\iota)})^{2(\Delta_\iota+1 \mod 2)} (-\beta_j^2) \right).$$

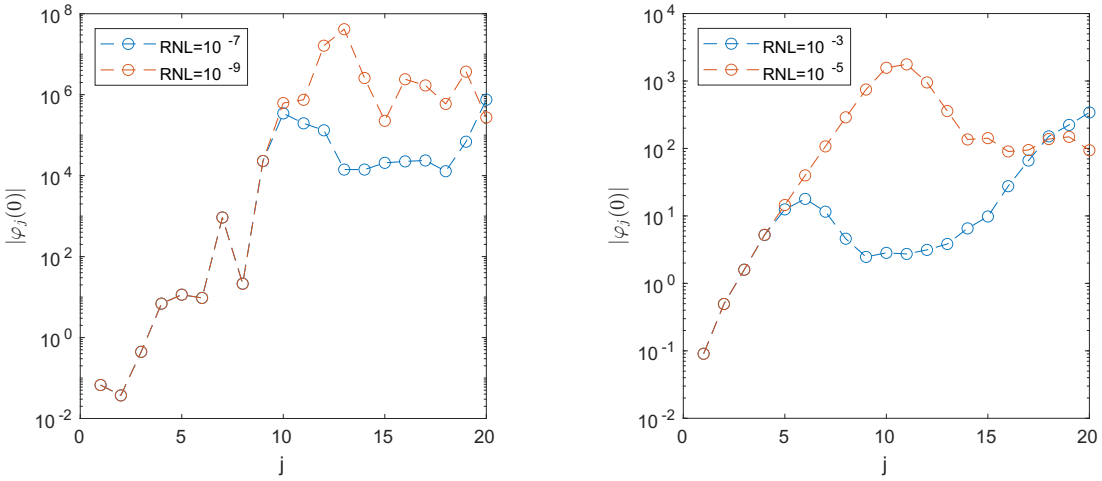
Protože $\mathcal{D}_j$ je sjednocením posloupností z $\mathcal{D}_{j-1}$, na jejichž konec přidáme 1, a posloupností z $\mathcal{D}_{j-2}$, na jejichž konec přidáme 2, je v důsledku

$$-\alpha_j N_{j-1} - \beta_j^2 N_{j-2} = \sum_{\{\Delta_\iota\}\in\mathcal{D}_j} \left(\prod_{\iota=1}^{\mu} (-1)^j \alpha_{S(\Delta_\iota)}^{(\Delta_\iota \mod 2)} (-\beta_{S(\Delta_\iota)})^{2(\Delta_\iota+1 \mod 2)} \right) = N_j.$$

□

[3]Autor této práce (jsa středoškolským učitelem) volí před žáky přístupnější pohled na tuto množinu: posloupnosti z $\mathcal{D}_j$ představují všechna možná vydláždění chodníku délky j dlaždicemi délky 1 nebo 2. Jak záhy ukážeme, stejnou strukturu mají i sčítance tvořící $\varphi_j(0)$

[4]V pomyslném dláždění odpovídá α_i dlaždici velikosti 1 na i-té pozici, zatímco β_i^2 dlaždici velikosti 2 na i-té a $i-1$-té pozici.



(a) `shaw(400)` *(b)* `gravity(400)`

Obrázek 3.1: Absolutní hodnota koeficientu $\varphi_j(0)$ pro dvě různé modelové úlohy o dvou různých RNL. Zdá se, že pro úlohu `shaw(400)` *je při $RNL = 10^{-7}$ $j_{rev} = 10$, při $RNL = 10^{-9}$ $j_{rev} = 13$, pro úlohu* `gravity(400)` *se pak při $RNL = 10^{-3}$ zdá být $j_{rev} = 6$ a při $RNL = 10^{-5}$ se jako j_{rev} jeví iterace 10 a nebo 11.*

Stejně jako v Golub-Kahanově bidiagonalizaci je tedy míra zanesení vektoru w_j šumem přímo závislá na velikosti koeficientu $\varphi_j(0)$, jehož analýza je v tomto případě ztížena tím, že je tvořen velkým počtem sčítanců.[5] To nám však nebrání jej v průběhu řešení průběžně sledovat, neboť jeho rekurzivní výpočet podle rekurence (3.5) je výpočetně nenáročný. Obrázek 3.1 ukazuje chování koeficientu $\varphi_j(0)$ pro různé hodnoty RNL. Vidíme, že oproti Golub-Kahanově bidiagonalizaci není v tomto případě $\varphi_j(0)$ monotónně rostoucí, do určité iterace však především roste, po jejím dosažení pak několik iterací stagnuje a nebo klesá. Čím vyšší je RNL, tím dříve je této iterace dosaženo. Stejně jako v Sekci 2.2 budeme tuto iteraci značit j_{rev}.

Obrázek 3.2 ukazuje, jak vypadají v konkrétní úloze jednotlivé vektory w_j. Po iteraci $j_{rev} = 10$ ve vektoru $w_{j_{rev}+1}$ opravdu převládá šumová složka a v jeho spektru jsou stejně zastoupeny všechny vektory u_ι pro $\iota > 11$.

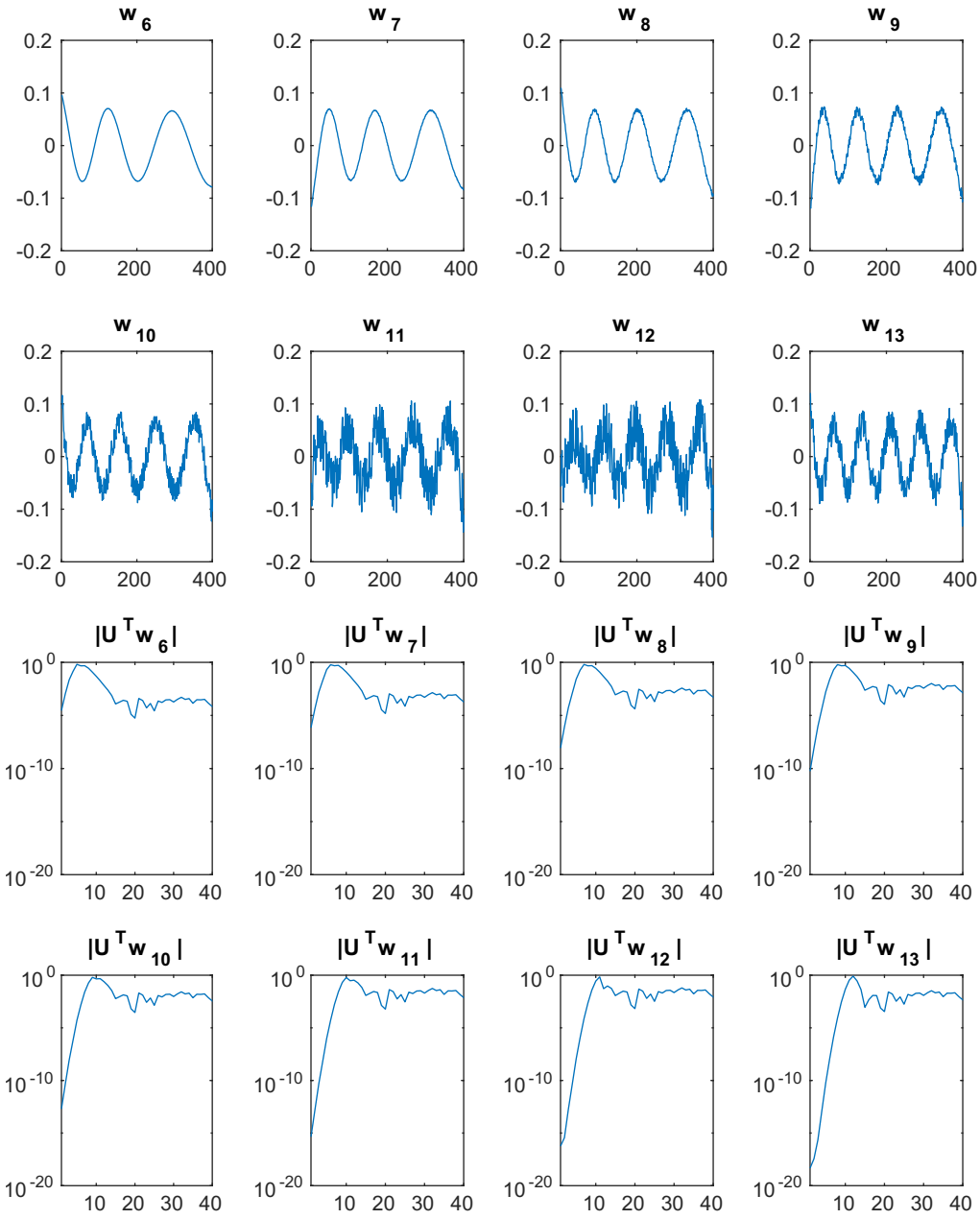
Pokusíme se nyní – analogicky jako [HPS09] – vysvětlit chování koeficientu $\varphi_j(0)$. Do rekurence (3.5) můžeme dosadit singulární rozklad symetrické matice $A = U\Sigma V^T$ a dostaneme rovnost

$$\begin{aligned} \beta_{j+1}U^T w_{j+1} &= \Sigma V^T w_j - \alpha_j U^T w_j - \beta_j U^T w_{j-1}. \\ \beta_2 U^T w_2 &= \Sigma V^T w_1 - \alpha_1 U^T w_1. \end{aligned}$$

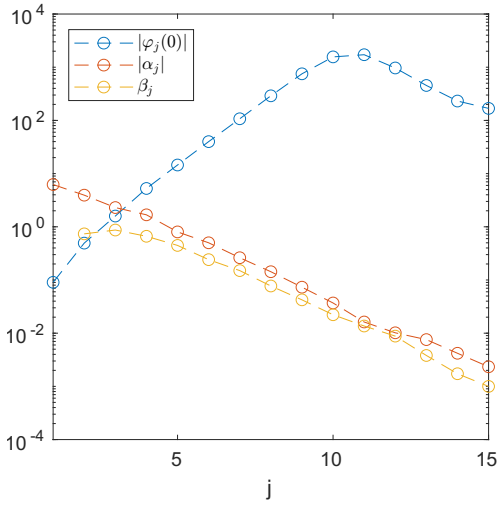
V první iteraci vidíme, že ve vektorech $\Sigma V^T w_1$ a $U^T w_1$ má tendenci převládat stejná spektrální komponenta a tato dominance je dokonce posílena přenásobením maticí Σ. Vektor w_2 je však oproti w_1 ortogonalizován, v $U^T w_2$ se proto musela dominantní komponenta z obou vektorů odečíst, a β_2 proto musí být relativně

[5] Počet sčítanců navíc exponenciálně narůstá, protože jak uvádí např. [MN02, 10.3],

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



Obrázek 3.2: Jednotlivé vektory w_j vygenerované Lanczosovým algoritmem pro úlohu `gravity(400)` *s $RNL = 10^{-5}$ a velikosti prvních 40 složek jejich spektra, tj. prvních 40 složek vektoru $|U^T s_j|$, kde U je matice ze singulárního rozkladu $A = U^T \Sigma V$. Již vektor w_{11} je masivně zanesen šumem, což odpovídá odhadu $j_{rev} = 10$ z Obrázku 3.1. Tomu odpovídá i jeho spektrum, vidíme, že jsou v něm všechny vektoru u_ι pro $\iota > 11$ zastoupeny rovnoměrně.*



Obrázek 3.3: Absolutní hodnoty koeficientů α_j, β_j a $\varphi_j(0)$ v úloze `gravity(400)` *s $RNL = 10^{-5}$. Je patrné, že pro $j = j_{rev} = 10$ zpomaluje pokles β_j, v důsledku čehož přestává růst $|\varphi_j(0)|$.*

menší než $|\alpha_1|$. Analogický proces probíhá v každé další iteraci: ve vektorech $U^T w_j$ a $\Sigma V^T w_j$ převažují stejné komponenty, k tomu, aby proti nim byl vektor $U^T w_{j+1}$ ortogonalizován, se v nich musejí tyto komponenty navzájem odečíst a β_j musí iteraci od iterace klesat, jmenovatel $\varphi_j(0)$ proto bude také klesat a celý koeficient v absolutní hodnotě poroste.

Po dosažení j_{rev} se situace změní tím, že ve vektoru $w_{j_{rev}}$ převažuje šum a není v něm tedy žádná dominantní komponenta – v důsledku toho předchozí úvaha přestává platit a růst $|\varphi_j(0)|$ se zastavuje. Detailněji toto chování analyzujeme v Sekci 4.1 s využitím periodogramů jednotlivých vektorů w_j.

3.3 Metoda minimálních reziduí a její varianty

V předchozích sekcích jsme popsali iterativní metodu, která vytváří ortonormální bázi $\mathcal{K}_j(A,b)$. Nyní popíšeme metody, které pomocí takto vytvořené báze aproximují řešení lineárního problému $Ax = b$ pro regulární a symetrickou matici A. Opět budeme postupovat iterativně: v každé iteraci budeme hledat vektor $x_j \approx x$, který leží v Krylovově prostoru $\mathcal{K}_j(A,b)$. Pro reziduum $r_j^{MINRES} = b - Ax_j$ proto musí platit

$$r_j^{MINRES} \in b + A\mathcal{K}_j(A,b).$$

Kdyby $x_j = x$, muselo by $r_j^{MINRES} = 0$. Chceme-li nalézt x_j, které je dobrou aproximací x, může být dobrou strategií hledat řešení ve smyslu nejmenších čtverců, tj. hledat takové x_j, aby $\|r_j^{MINRES}\|$ byla co nejmenší. Metodu, v níž takové x_j hledáme, nazýváme *metoda minimálních reziduí*, zkráceně MINRES, a poprvé byla popsána v článku [PS75].

Abychom požadavek na minimalitu rezidua naplnili, rozložíme si r_j^{MINRES} na součet

$$r_j^{MINRES} = r_j^{MINRES}|_{A\mathcal{K}_j(A,b)} + r_j^{MINRES}|_{A\mathcal{K}_j(A,b)^\perp}.$$

Minimální normy $\|r_j^{MINRES}\|$ pak dosáhneme tak, že $r_j^{MINRES}|_{A\mathcal{K}_j(A,b)} = 0$, tj.

$r_j^{MINRES} \perp A\mathcal{K}_j(A,b)$. V každé iteraci tedy budeme hledat x_j splňující

$$x_j \in \mathcal{K}_j(A,b),\ b - Ax_j = r_j^{MINRES} \perp A\mathcal{K}_j(A,b).$$

Protože platí, že

$$\|r_j\| = (b - Ax_j)^T(b - Ax_j) = (Ax - Ax_j)^T(Ax - Ax_j) = \\ = (x - x_j)^T A^T A(x - x_j) = \|x - x_j\|_{A^TA},$$

je řešení x_j optimální v tom smyslu, že má nejmenší A^TA-normu chyby. Popíšeme nyní obecný algoritmus, jak tuto aproximaci najít.

Položme na úvod $w_1 := b/\|b\|$. Pomocí Lanczosova algoritmu vygenerujeme ortonormální bázi $\{w_1,w_2,\dots w_{j+1}\}$ Krylovova prostoru $\mathcal{K}_j(A,b)$. Nyní budeme hledat řešení x_j tvaru $x_j = W_j y_j$, kde y_j je vektor koeficientů x_j v bázi $\mathcal{K}_j(A,b)$. Tento vektor y_j bývá nazýván *vnitřní řešení* metody MINRES. Reziduum r_j^{MINRES} můžeme vyjádřit pomocí vztahu (3.4) jako

$$r_j^{MINRES} = b - Ax_j = b - A(W_j y_j) = b - AW_j y_j = b - AW_j y_j = \\ = \|b\|w_1 - W_{j+1}T_{j+}y_j = W_{j+1}(\|b\|e_1 - T_{j+}y_j), \quad (3.10)$$

minimalizujeme-li jeho normu, musí platit

$$\|r_j^{MINRES}\| = \|W_{j+1}(\|b\|e_1 - T_{j+}y_j)\| = \min_{y\in\mathbb{R}^j}\|\|b\|e_1 - T_{j+}y\|. \quad (3.11)$$

Vektor y_j proto můžeme najít pomocí metody nejmenších čtverců uplatněné na výrazně menší úlohu než byl původní problém $Ax \approx b$.

Díky tomu, že je matice A symetrická, je možné počítat vektory w_j za pomoci krátkých rekurencí. Řešení x_j lze tedy nalézt bez potřeby udržovat v paměti celou bázi $\mathcal{K}_j(A,b)$ – je však potřeba jej vytvářet iterativně za pomoci vhodně použitých Givensových rotací, efektivní implementace popsaná např. v [Saa03, Sekce 6.5.3] je proto relativně složitá.

Metodu MINRES popsanou v této sekci můžeme ještě mírně pozměnit. Předpokládejme nyní, že ještě před zahájením výpočtu máme k dispozici odhad řešení x_0, od nějž můžeme začít aproximovat. V takovém případě můžeme položit $r_0^{MINRES} = b - Ax_0$ a řešení x_j hledat ve varietě $x_0 + \mathcal{K}_j(A,r_0^{MINRES})$. Pro r_j^{MINRES} pak bude platit

$$r_j^{MINRES} = r_0^{MINRES} + A\mathcal{K}_j(A,r_0^{MINRES}).$$

Samotný výpočet nám však tento předpoklad změní jen lehce: namísto $w_1 = b/\|b\|$ položíme $w_1 = r_0^{MINRES}/\|r_0^{MINRES}\|$ a skutečnost, že y_j minimalizuje $\|r_j^{MINRES}\|$ znamená, že

$$y_j = \operatorname{argmin}_{y\in\mathbb{R}^j}\|\|r_0^{MINRES}\|e_1 - T_{j+}y\|.$$

Chyba metody MINRES s pravou stranou zanesenou šumem

Pokud metoda MINRES dostane na vstupu vektor b prostý šumu, měli bychom (zanedbáme-li zaokrouhlovací chyby) dostat iteraci od iterace přesnější řešení –

norma $\|x - x_j\|_{A^TA}$ je klesající funkcí j. V případě, že pravou stranu b zaneseme šumem, budeme v metodě MINRES pozorovat stejné chování, jako jsme viděli u LSQR. Před dosažením j_{rev} bude norma chyby $\|x - x_j\|$ typicky klesat, v okamžiku dosažení j_{rev} převládne ve vektorech w_ι pro $\iota \geq j_{rev}$ šum, který se v dalších iteracích rozšíří i do vektoru x_j a norma chyby tedy od dosažení j_{rev} poroste. Tuto semikonvergenci ilustruje Obrázek 3.4a. V okamžiku dosažení iterace j_{rev} tedy pravděpodobně dostaneme nejpřesnější aproximaci x_j a je na místě v tuto chvíli zastavit výpočet. Otevřenou otázkou, která dává prostor dalšímu bádání, jsou metody detekce j_{rev}. Propagaci šumu do řešení x_j dále ilustruje Sekce 4.2.

Stojí za povšimnutí, že minimální chybu dostáváme právě pro maximální $|\varphi_j(0)|$ z Obrázku 3.1. To je rozdíl oproti metodě LSQR, kde minimum chyby následovalo až několik iterací po j_{rev}. V metodě LSQR jsme totiž vyjadřovali řešení pomocí vektorů w_j, zatímco šum se propagoval primárně ve vektorech s_j, takže po dosažení j_{rev} ještě pár iterací zabrala propagace chyby do druhé posloupnosti vektorů.

Metoda MR-II

Slabinou metody MINRES při řešení úloh s pravou stranou zatíženou šumem je zanesení Krylovova prostoru $\mathcal{K}_j(A,b)$ šumem z vektoru b. Řešením je pracovat s Krylovovým prostorem $\mathcal{K}_j(A,Ab)$, jehož vektory jsou přenásobeny maticí A a tím pádem zhlazeny. Tato metoda se nazývá MR-II [Han95, 6.1], původně byla vytvořena pro řešení lineárních systémů se singulární maticí a pro praktické použití k řešení diskrétních inverzních úloh je výrazně vhodnější než původní MINRES.

Analýza, kterou jsme provedli v Sekci 3.2, lze vztáhnout i na metodu MR-II. Konstruujeme-li bázi Krylovova prostoru $\mathcal{K}_j(A,Ab)$, platí tvrzení zcela analogické k Lemmatu 3.6: vektor w_{j+1} z báze takového Krylovova prostoru můžeme vyjádřit jako

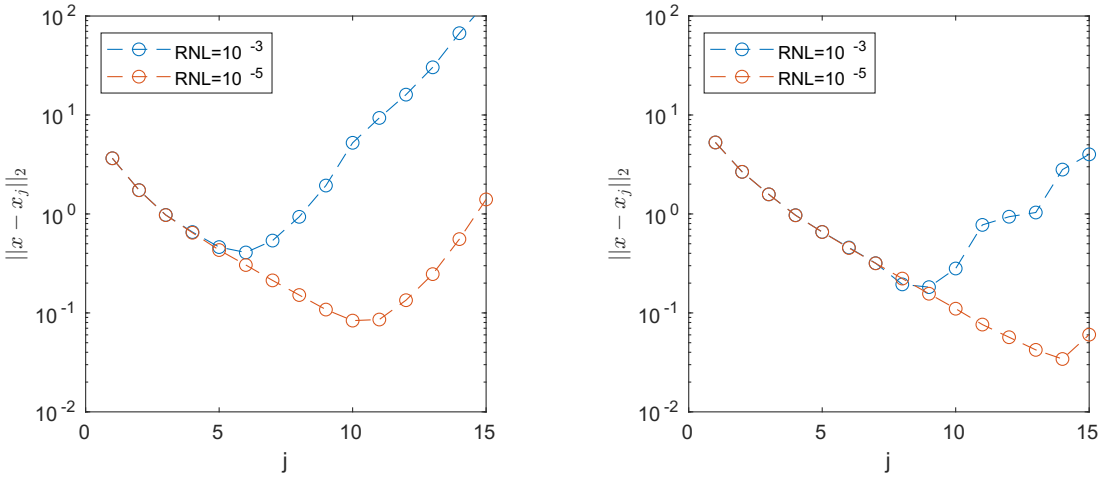
$$w_{j+1} = \varphi_j(A)w_1 = \varphi_j(A)Ab,$$

kde polynom φ_j splňuje stejné podmínky jako v Lemmatu 3.6.

Do této rovnosti můžeme opět zcela analogicky dosadit $Ab = A^2x - Ae$ a analogicky jako v Lemmatu 13 můžeme vektor w_{j+1} rozdělit na vektor přenásobený dokonce maticí A^2, a na vektor $\varphi_j(0)Ae$, který je přenásoben pouze maticí A. Matice A sice nepotlačí veškerý šum, výrazně však sníží jeho hladinu, proto k projevení šumu dojde až v pozdější iteraci a tím pádem získáme výrazně přesnější řešení (což vidíme na obrázcích 3.4a a 3.4b). Numerické chování vektorů w_j a koeficientu $\varphi_j(0)$ v metodě MR-II shrnuje Sekce 4.3. Detailnější analýzu a porovnání metod MINRES a MR-II provádí [JH07].

Metoda GMRES

Až dosud jsme v této sekci předpokládali, že matice A bude symetrická. V případě, že z tohoto předpokladu slevíme a zobecníme náš problém na A libovolnou čtvercovou regulární, metodu MINRES stačí mírně modifikovat. Jediné místo, kde jsme pracovali s předpokladem symetrie, jsou rovnosti (3.10) a (3.11). Nesymetrická matice A totiž nesplňuje vztah $AW_j = W_{j+1}T_{j+}$, kde T_{j+} je rozšířená



(a) MINRES *(b) MR-II*

Obrázek 3.4: Srovnání normy chyby v jednotlivých iteracích metod MINRES a MR-II pro modelovou úlohu `gravity(400)` *se dvěma různými úrovněmi šumu.*

Jacobiho matice, ale jen $AW_j = W_{j+1}H_{j+}$, kde H_{j+} je rozšířená horní Hessenbergova matice. V důsledku toho však nemůžeme počítat vektory w_j pomocí krátkých rekurencí, každý vektor Aw_j musíme ortogonalizovat proti všem předchozím, a musíme proto udržovat v paměti celou bázi $\mathcal{K}_j(A,b)$. Výsledná metoda se nazývá *zobecněná metoda minimálních reziduí*, zkráceně GMRES, a poprvé byla publikována v článku [SS86]. Tato metoda má s metodou MINRES řadu vlastností společnou, často bývá studována rovnou jako obecnější algoritmus. Pro analýzu, které se věnuje tato práce, je však problémem tvorba ortonormální báze prostoru $\mathcal{K}_j(A,b)$, která vyžaduje Arnoldiho algoritmus s dlouhými rekurencemi. Vyjádření koeficientu, jímž je v této metodě přenásoben šum, je proto výrazně náročnější, explicitní vzorec odvozuje Sekce 3.5, podrobnější studium jeho chování je však mimo možnosti této práce. I metodu GMRES lze modifikovat tak, že budeme vyhledávat vektor x_j v Krylovově prostoru $\mathcal{K}_j(A,Ab)$, tato varianta se nazývá RRGMRES, je představena v [CLR00] a i její analýze se věnuje [JH07].

3.4 Norma rezidua v metodě MINRES

V této sekci odvodíme, jak závisí norma rezidua metody MINRES na koeficientu $\varphi_j(0)$. Postupovat budeme analogicky jako v Sekci 2.3 – na úvod budeme pracovat s reziduem jednodušší metody, konkrétně metody FOM. Zkratka FOM znamená *full orthogonalization method*, česky též *metoda ortogonálních reziduí*, detailně je metoda popsaná v [Saa03, 6.4]. Tato metoda postupuje podobně jako MINRES, najde ortonormální bázi $\{w_1,w_2,\ldots,w_j\}$ Krylovova prostoru $\mathcal{K}_j(A,b)$. Rozdíl je v tom, co v ní musí splňovat řešení. Pro řešení x_j^{FOM} musí platit

$$b - Ax_j^{FOM} \perp \mathcal{K}_j(A,b).$$

Za předpokladu, že matice T_j je regulární[6], můžeme řešení x_j^{FOM} nalézt pomocí vztahu

$$x_j^{FOM} = W_j y_j^{FOM}$$
$$T_j y_j^{FOM} = \|b\| e_1,$$

Z tohoto vztahu můžeme snadno dovodit následující lemma [Saa03, Tvrzení 6.7], s jehož pomocí dokážeme nalézt souvislost mezi normou rezidua a $\varphi_j(0)$:

Lemma 15. *Pokud aproximace x_j^{FOM} existuje, pak je reziduum metody FOM rovno*

$$r_j^{FOM} = b - Ax_j^{FOM} = \frac{1}{\varphi_j(0)} w_{j+1}.$$

Důkaz. Podle definice je reziduum r_j^{FOM} rovno

$$b - Ax_j^{FOM} = b - AW_j y_j^{FOM} = b - AW_j y_j^{FOM} = \\ = \|b\| w_1 - W_j T_j y_j^{FOM} - \beta_{j+1} e_j^T y_j w_{j+1} = \Phi_j(A) b \quad (3.12)$$

pro polynom $\Phi_j \in \mathcal{P}_j$ s absolutním členem $\Phi_j(0) = 1$. Protože $T_j y_j^{FOM} = \|b\| e_1$, platí i rovnost $W_j T_j y_j^{FOM} = \|b\| w_1$ a rovnice (3.12) se redukuje na vztah $r_j^{FOM} = -\beta_{j+1} e_j^T y_j w_{j+1}$. Podle Lemmatu 12 je proto

$$r_j^{FOM} = -\beta_{j+1} e_j^T y_j \varphi_j(A) b.$$

Srovnáme-li tyto informace, zjistíme, že $-\beta_{j+1} e_j^T y_j \varphi_j(A) b = \Pi_j(A) b$. Tato rovnost tím pádem musí platit i pro absolutní členy obou polynomů, dostáváme proto $-\beta_{j+1} e_j^T y_j \varphi_j(0) b = \Pi_j(0) b = 1$. Tím pádem se musí být koeficient $-\beta_{j+1} e_j^T y_j$ roven $\varphi_j(0)^{-1}$, z čehož vyplývá dokazovaná rovnost. □

Tento výsledek je pro nás klíčový v tom, že [Saa03, 6.5.7] popisuje závislost normy $\|r_j^{FOM}\|$ a normy rezidua $\|r_j^{MINRES}\|$, které je produktem metody MINRES. Z této závislosti můžeme vyvodit následující větu.

Věta 16. *Norma rezidua $r_j^{MINRES} = Ax_j - b$ je rovna*

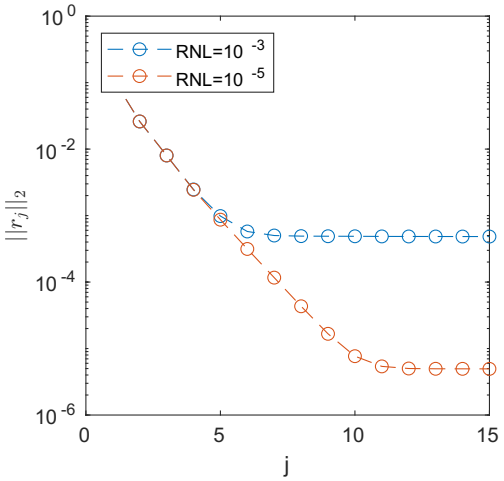
$$\|r_j^{MINRES}\| = \frac{1}{\sqrt{\sum_{\iota=1}^{j} \varphi_\iota(0)^2}}$$

Důkaz. Tato věta je přímým důsledkem [Saa03, Věta 6.14]: platí, že

$$\|r_j^{MINRES}\| = \frac{1}{\sqrt{\sum_{\iota=1}^{j} \|r_\iota^{FOM}\|^{-2}}} = \frac{1}{\sqrt{\sum_{\iota=1}^{j} \varphi_\iota(0)^2}}.$$

□

[6]Matice T_j může být obecně v některém kroku j singulární a řešení x_j^{FOM} tedy nemusí existovat. Vzhledem k tomu, že vlastní čísla dvou po sobě jdoucích Jacobiho matic se navzájem ostře prokládají [DTHPS12, Cvičení 7.6], [HJ85, Sekce 4.3], bude matice T_{j+} opět regulární a řešení x_{j+1}^{FOM} existovat bude.



Obrázek 3.5: Průběh normy rezidua metody LSQR pro úlohu `gravity(400)` *pro dvě různé hladiny šumu. Vidíme, že do dosažení iterace j_{rev} norma rezidua klesá, po jejím dosažení již jen stagnuje.*

Z Věty 16 proto vyplývá[7], že reziduum metody MINRES je nerostoucí funkcí j, což je logické – minimalizujeme jej přes čím dál větší prostor $\mathcal{K}_j(A,b)$. Z analýzy v Sekci 3.2 navíc vyplývá, že do dosažení iterace j_{rev} budou sčítance $\varphi_\iota(0)^2$ stabilně narůstat a v normě rezidua tedy budeme dělit čím dál větším součtem. Po dosažení j_{rev} však poklesnou a norma rezidua bude již spíše stagnovat. Tento fakt ilustruje i Obrázek 3.5.

3.5 Šíření šumu v Arnoldiho algoritmu

V Sekci 3.2 jsme nalezli explicitní vyjádření koeficientu, jímž je přenásoben šum ve vektorech w_j generovaných Lanczosovým algoritmem. Již toto vyjádření je relativně těžkopádné a pro praktickou implementaci je mnohem vhodnější rekurentní vyjádření koeficientu $\varphi_j(0)$. V této sekci ukážeme odvození analogického koeficientu pro Arnoldiho algoritmus.

Pro vektor w_{j+1} generovaný Arnoldiho algoritmem můžeme z rekurence (3.1) vyvodit následující lemma, analogické k Lemmatům 6 a 12:

Lemma 17. *Vektor w_{j+1} generovaný Arnoldiho algoritmem můžeme vyjádřit jako*

$$w_{j+1} = \varphi_j(A)b,$$

kde $\varphi_j \in \mathcal{P}_j$ je polynom stupně j, který splňuje rekurenci

$$\begin{aligned} h_{j+1,j}\varphi_j(A) &= (A - h_{j,j}I)\varphi_{j-1}(A) - \sum_{i=1}^{j-1} h_{i,j}\varphi_{i-1}(A), \\ \varphi_0(A) &= \|b\|^{-1}. \end{aligned} \tag{3.13}$$

[7] Přestože x_j^{FOM} a tedy ani r_j^{FOM} nemusí pokaždé existovat, tato věta platí pro všechna j. Jak ukazuje [Saa03, Věta 6.17], v případě, že T_j je singulární a neexistuje x_j^{FOM}, metoda MINRES stagnuje a její reziduum tím pádem zůstává definované, jen se nemění a reziduum r_j^{FOM} v takovém případě nepřičítáme.

I zde jsou všechny koeficienty polynomu $\varphi_j(A) - \varphi_j(0)$ přenásobené maticí A. Můžeme tedy vyslovit následující lemma, analogické k Lemmatu 13.

Lemma 18. *Vektor w_{j+1} generovaný Arnoldiho algoritmem můžeme rozložit na součet*

$$w_{j+1} = w_{j+1}^{LF} + \varphi_j(0)e,$$

kde $w_{j+1}^{LF} = \varphi_j(A)(Ax - e) = \varphi_j(A)Ax + [\varphi_j(A) - \varphi_j(0)]e$.

Celý vektor w_{j+1}^{LF} je přenásobený maticí A, a proto jsou v něm potlačeny hladké frekvence. Na posílení šumu ve vektoru w_{j+1} má proto zásadní vliv absolutní člen polynomu φ_j. Dosadíme-li nyní nulu do rekurence (3.13), dostaneme pro absolutní člen $\varphi_j(0)$ rekurenci

$$\begin{aligned} h_{j+1,j}\varphi_j(0) &= -h_{j,j}\varphi_{j-1}(0) - \sum_{i=1}^{j-1} h_{i,j}\varphi_{i-1}(0) = -\sum_{i=1}^{j} h_{i,j}\varphi_{i-1}(0), \\ \varphi_0(0) &= \|b\|^{-1}. \end{aligned} \tag{3.14}$$

I z této rovnosti můžeme přímo vypočítat $\varphi_j(0)$ pro libovolné $j \in \mathbb{N}$, výsledný vzorec je však ještě složitější než u Lanczosova algoritmu. Opět budeme muset definovat pomocnou množinu, označme ji tentokrát $\mathcal{P}_j$.

Definice 7. *Množinu všech permutací na j prvcích $\pi \in \mathcal{S}_j$ takových, že $\pi(\iota) \geq \iota - 1$ pro všechna $\iota \in \{1, \ldots, j\}$, označíme $\mathcal{P}_j$.*

Vidíme, že $|\mathcal{P}|_j = 2^{j-1}$, neboť pro číslo j máme dva možné obrazy j a $(j-1)$, pro číslo $(j-1)$ nám v důsledku zbývají také dva možné obrazy (vybíráme mezi $(j-2)$,$(j-1)$ a j s tím, že jedno z těchto čísel je již obsazeno) a stejným způsobem máme dva možné obrazy pro všechna čísla až do 2, číslo 1 již má svůj obraz daný jednoznačně.

Pro jednodušší vyjádření koeficientu $\varphi_j(0)$ se nám bude hodit rozšířit matici H_{j+}. Definujeme proto matici H_{j++} jako

$$H_{j++} = \begin{pmatrix} e_1 \\ H_{j+} \end{pmatrix}.$$

Lemma 19. *Pro $j < st_b(A)$ je matice H_{j++} regulární.*

Důkaz. Matice H_{j++} je čtvercová, ukážeme, že má nenulový determinant. Protože je zároveň dolní trojúhelníková, je její determinant roven součinu prvků na hlavní diagonále. Protože $j < st_b(A)$, budou $h_{i+1,i} \neq 0$ a tedy $\det(H_{j++}) \neq 0$. □

Věta 20. *Koeficient $\varphi_j(0)$ je pro Arnoldiho algoritmus roven*

$$\varphi_j(0) = \frac{(-1)^j \det(H_j)}{\det(H_{j++})} = \frac{(-1)^j}{\prod_{i=1}^{j+1} h_{i,i-1}} \sum_{\pi \in \mathcal{P}_j} \operatorname{sgn}(\pi) \prod_{i=1}^{j} h_{i,\pi(i)}. \tag{3.15}$$

Důkaz. Víme, že $\varphi_0(0) = \|b\|^{-1}$. Přičteme-li k rovnosti (3.14) sumu z pravé strany, dostáváme pro každé $\iota \leq j$ rovnost

$$\sum_{i=1}^{\iota+1} h_{i,\iota}\varphi_i(0) = 0. \tag{3.16}$$

Koeficienty $\varphi_\iota(0)$ pro $\iota \leq j$ můžeme shrnout do jednoho vektoru f_j tvaru

$$f_j = \begin{pmatrix} \varphi_0(0) \\ \varphi_1(0) \\ \vdots \\ \varphi_j(0) \end{pmatrix}$$

a rovnost (3.16) můžeme zapsat maticově jako

$$H_{j++}^T f_j = \|b\|^{-1} e_1. \tag{3.17}$$

Matice H_{j++} je regulární, koeficient $\varphi_j(0)$ můžeme proto vyjádřit z rovnice (3.17) pomocí Cramerova pravidla [BT18, Věta 7.28]:

$$\varphi_j(0) = \frac{\det(X)}{\det(H_{j++})^T},$$

přičemž matice X je tvaru

$$X = \begin{pmatrix} 1 & 0 & 0 & 0 & \dots & \|b\|^{-1} \\ h_{1,1} & h_{2,1} & & & & 0 \\ h_{1,2} & h_{2,2} & h_{3,2} & & & 0 \\ \vdots & \vdots & \vdots & \ddots & & \vdots \\ h_{1,j} & h_{2,j} & h_{3,j} & \dots & h_{j,j} & 0 \end{pmatrix}.$$

Rozvineme-li $\det(X)$ podle prvního řádku, získáme $\det(X) = (-1)^{j+1+1} \det(H_j)$. Jelikož transpozice nemění determinant, je $\varphi_j(0)$ rovno

$$\varphi_j(0) = \frac{\det(X)}{\det(H_{j++}^T)} = \frac{(-1)^j \det(H_j)}{\det(H_{j++})} = \frac{(-1)^j}{\prod_{i=1}^{j+1} h_{i,i-1}} \sum_{\pi \in \mathcal{P}_j} \operatorname{sgn}(\pi) \prod_{i=1}^{j} h_{i,\pi(i)},$$

přičemž poslední rovnost vyplývá z definice determinantu a z tvaru matice H_j. □

Získali jsme tedy explicitní vzorec pro koeficient $\varphi_j(0)$, který určuje propagaci šumu ve vektorech w_j generovaných Arnoldiho algoritmem. Vidíme, že výpočet tohoto koeficientu přímo je dosti složitý, při praktické implementaci Arnoldiho algoritmu je mnohem jednodušší tento koeficient počítat rekurentně podle (3.14). Analýza chování tohoto koeficientu je v důsledku (3.15) poměrně obtížná a přesahuje rozsah a zaměření této práce.

4. Numerické experimenty

Tato kapitola obsahuje numerické experimenty, které ilustrují jevy popsané v Kapitole 2 a 3. Všechny experimenty v této i předchozích kapitolách byly provedeny v prostředí Matlab, verzi R2023a, na počítači vybaveném procesorem Intel Core i5-4460, s využitím testovacích matic a skriptů z balíčku Regularization Tools [Han07]. Algoritmy Golub-Kahanovy iterační bidiagonalizace, Lanczosova a Arnoldiho procesu si autor implementoval sám, stejně jako metodu LSQR a zjednodušené verze algoritmů MINRES a MR-II (vzhledem k účelu práce však tyto algoritmy po celou dobu uchovávají celou bázi W_j).

Není-li řečeno jinak, veškeré výpočty ortogonální báze v Golub-Kahanově bidiagonalizaci a v Lanczosově algoritmu byly provedeny s reortogonalizací – namísto výpočtu pomocí krátkých rekurencí byl každý vektor s_j a w_j ortogonalizován vůči všem předchozím. Cílem bylo co nejvěrněji simulovat výpočet v přesné aritmetice – v praxi však reortogonalizace kvůli vysokým výpočetním a paměťovým nákladům není prováděna, dochází tedy ke zpoždění konvergence metody k naivnímu řešení a propagace šumu (detaily popisujeme v Sekci 4.4),

4.1 Vztah koeficientů a bázových vektorů v Lanczosově algoritmu

V Sekci 3.2 jsme popsali chování koeficientu $\varphi_j(0)$ a jím zatížených vektorů w_j pro konkrétní zadání úlohy `gravity(400)` s $RNL = 10^{-5}$. V této sekci ilustrujeme šíření šumu i pro další problém: již zmíněnou úlohu `shaw(400)` s $RNL = 10^{-7}$. Obrázek 4.1 ukazuje hodnoty $\varphi_j(0), \alpha_j$ a β_j pro různá j. Oproti úloze `gravity(400)` zde není vývoj těchto koeficientů ani zdaleka monotónní – pozoruhodná je zejména iterace $j = 8$, v níž dochází k jednorázovému poklesu všech tří hodnot. Mimo to si můžeme také povšimnout, že v této iteraci navíc stagnuje norma chyby této metody. Z obou obrázků je dobře patrné, že $j_{rev} = 10$.

Obrázek 4.2 pak ukazuje vektory w_j, jejich spektra a navíc i *normalizované kumulativní periodogramy*. Normalizovaný kumulativní periodogram (zkráceně NCP) je diagram, který reprezentuje zastoupení různých frekvencí v daném vektoru. Pro vektor w_j nejprve spočítáme diskrétní Fourierovu transformaci

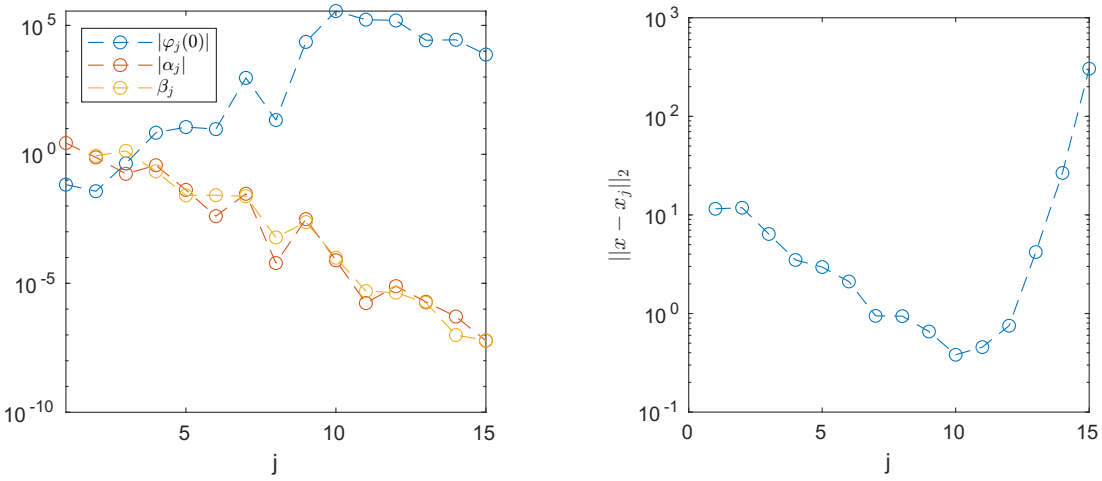
$$\hat{w_j} = \Big((\hat{w_j})_1, (\hat{w_j})_2, \ldots, (\hat{w_j})_n\Big)^T$$

(prostřednictvím algoritmu FFT [BS17, 8.2] je to výpočetně nenáročné), a následně definujeme vektor $c(w_j)$ s $q = \lfloor n/2 \rfloor$ složkami pomocí vztahu

$$c(w_j)_i = \frac{\sum_{\iota=1}^{i+1} |(\hat{w_j})_\iota|^2}{\sum_{\iota=1}^{q+1} |(\hat{w_j})_\iota|^2}.$$

Při prvním pohledu na vektor w_8 se zdá, že k anomálii z Obrázku 4.1 není důvod – šum se ve vektoru na pohled neprojevil, jeho periodogram je na pohled úplně stejný, jako u vektorů w_7 a w_9. Rozdíl je však dobře patrný ve spektru.

Vektory $U^T w_j$ pro $j < j_{rev}$ mají dominantní j-tou komponentu, předchozí komponenty jsou blízké nule, protože w_j jsou navzájem ortogonalizovány. Následující komponenty jsou oproti j-té velmi malé. Velikost těchto komponent však ve



Obrázek 4.1: Hodnoty $|\varphi_j(0)|, |\alpha_j|$ a β_j z Lanczosova algoritmu a normy chyby metody MINRES pro úlohu `shaw(400)` *s* $RNL = 10^{-7}$

vektoru $U^T w_8$ skokově vzroste o několik řádů – zdá se, že v této iteraci došlo k výraznému odečtení dominantních složek vektoru w_j, v důsledku čehož signifikantně pokleslo i $|\alpha_j|$ a β_j.

Jak však ukazuje normalizovaný kumulativní periodogram, ve vektoru w_8 stále ještě převládá dominantní komponenta nad šumem. To se mění až ve vektoru w_{11}, v němž jsou všechny frekvence zastoupeny stejně a tento vektor je již na první pohled tvořen především šumem.

4.2 Vnitřní řešení metody MINRES

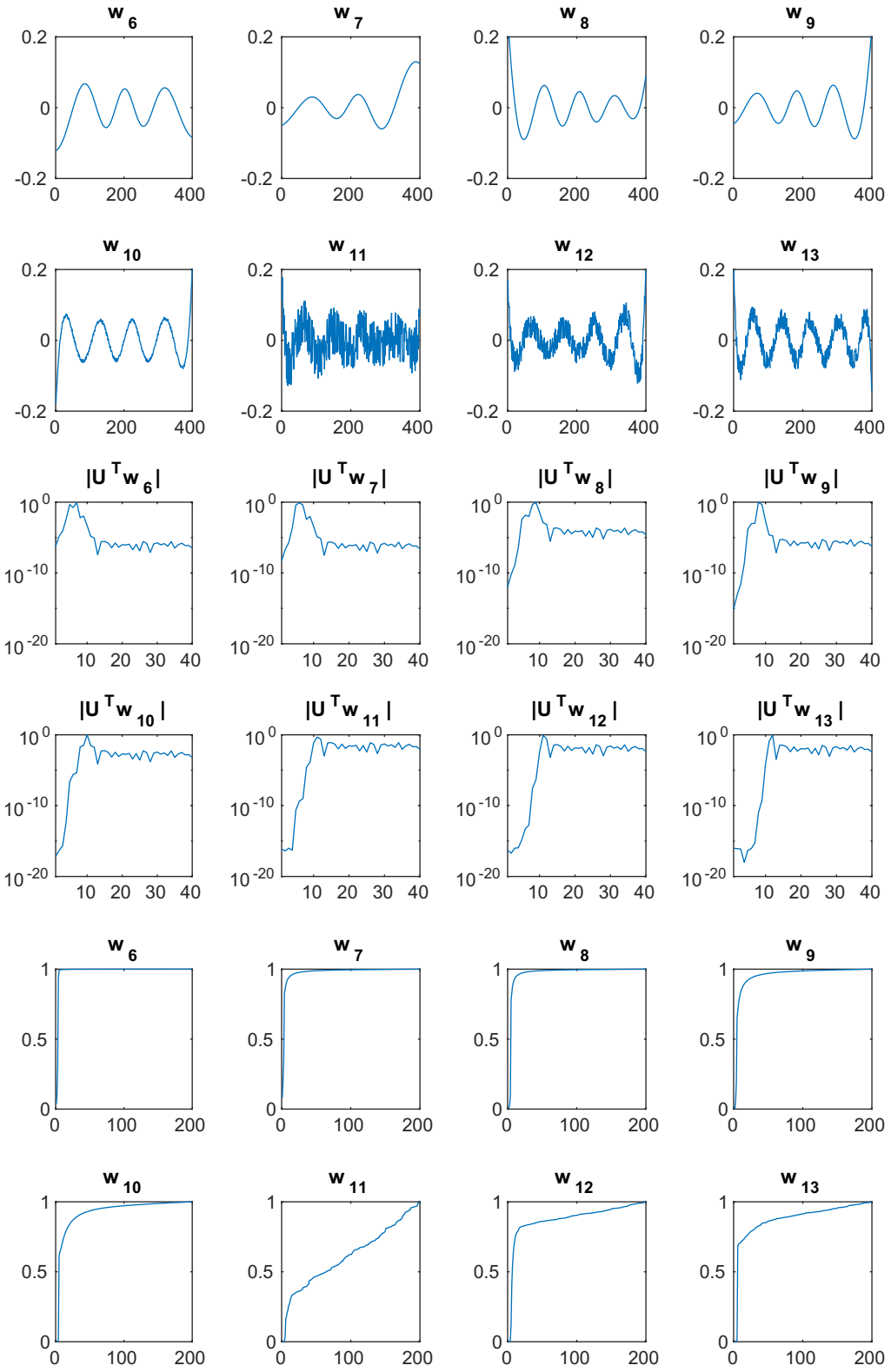
Ze Sekcí 3.2 a 3.4 již víme, jaký vliv má iterace j_{rev} na chování koeficientu $\varphi_j(0)$ v Lanczosově algoritmu a na normu rezidua r_j^{MINRES} pro aproximaci x_j získanou z metody MINRES. V předchozích sekcích jsme na tomto základě vyvodili, jak bude vypadat norma odpovídající chyby $\|x^{exact} - x_j\|$. Zaměřme se nyní na zkoumání toho, jak tato chyba vzniká a jak se do x_j dostává šum z vektorů w_ι pro $\iota \leq j$.

Podle rovnic (3.10) a (3.11) je

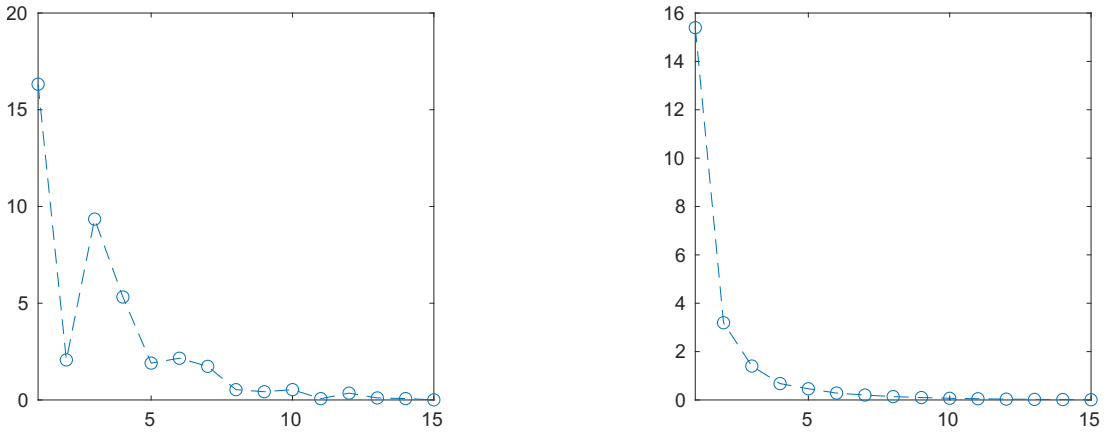
$$x_j = W_j y_j, \qquad y_j = \min_{y \in \mathbb{R}^j} \|\|r_0\| e_1 - T_{j+} y\|.$$

O tom, jak se bude řešením x_j šířit šum, proto rozhodují složky vektoru y_j. Přesné řešení x^{exact} bývá v diskrétních inverzních úlohách hladké. V případě, že by pravá strana b nebyla zanesena šumem, bychom proto měli tvořit x_j především z hladkých vektorů, absolutní hodnoty složek y_j by proto měly s rostoucím j klesat k nule. Tento jev ilustruje Obrázek 4.3.

V případě, že pravou stranu zatížíme šumem, ale metoda MINRES pouze semikonverguje – od určité iterace začne ve vektoru x_j převažovat šum. V testovací úloze `shaw(400)` nyní spočítáme y_j a pokusíme se toto chování vysvětlit. Jak ukazuje Obrázek 4.4, nejen že se počínaje $j = j_{rev}$ propaguje šum ve vektorech w_j, ale ve vnitřním řešení y_j posilují právě složky odpovídající vektoru $w_{j_{rev}}$ a několika okolním, které jsou šumem zaneseny nejvíce (z Obrázků 3.2 a 4.2 vidíme, že další vektory w_j pro $j > j_{rev}$ jsou opět alespoň částečně zhlazeny).

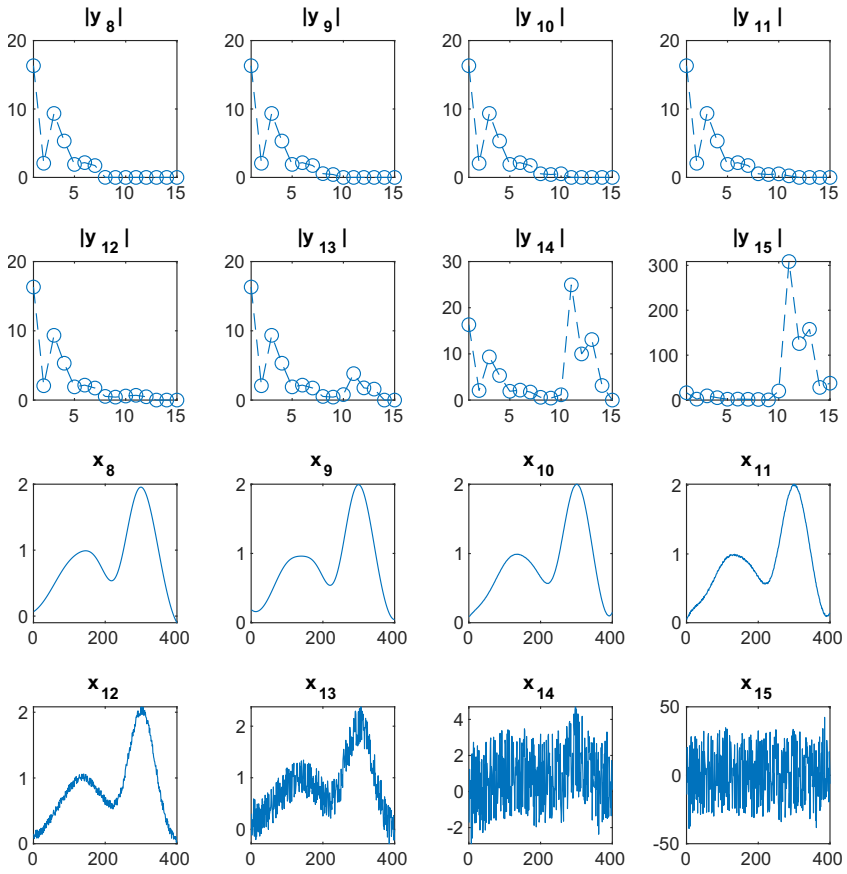


Obrázek 4.2: Vektory w_i spočtené Lanczosovým algoritmem pro úlohu `shaw(400)` *s* $RNL = 10^{-7}$*, prvních 40 složek jejich spektra a jejich normalizované kumulativní periodogramy.*

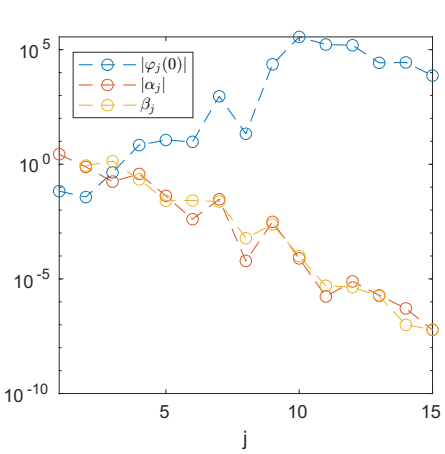


(a) `shaw(400)` *(b)* `gravity(400)`

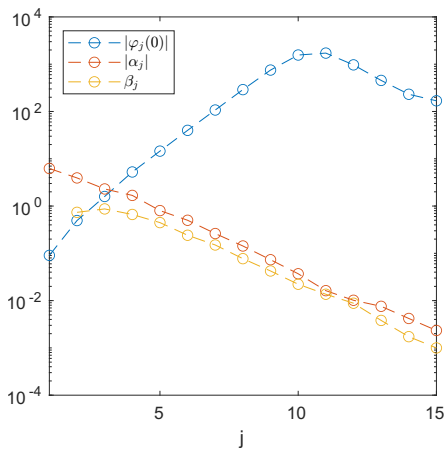
Obrázek 4.3: Absolutní hodnoty složek vektoru y_{15} získaného metodou MINRES pro dvě standardní úlohy s pravou stranou b nezatíženou šumem. Vidíme, že tyto složky nemusejí klesat monotónně, ale že se v obou případech postupně blíží nule.



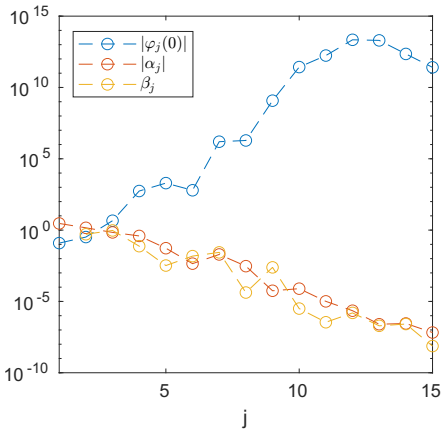
Obrázek 4.4: Absolutní hodnoty složek vnitřního řešení y_j a odpovídající aproximace řešení x_j pro úlohu `shaw(400)` *s pravou stranou zatíženou šumem s $RNL = 10^{-7}$. Z Obrázku 4.1 víme, že pro tuto úlohu je v Lanczosově algoritmu $j_{rev} = 10$. Od této iterace se ve vektorech y_j posiluje 10. až 12. složka, která do výsledného řešení zanáší vektory w_{10}, w_{11} a w_{12}, které jsou (jak je patrné z Obrázku 4.2) silně zatíženy šumem.*



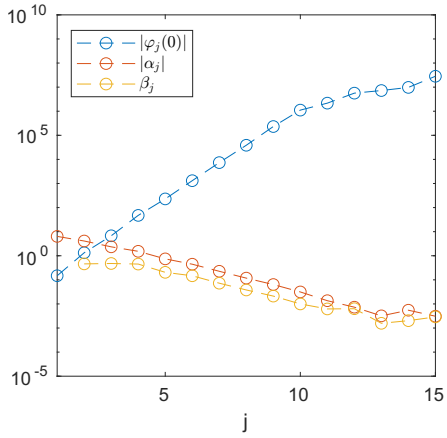
(a) MINRES pro `shaw(400)`



(b) MINRES pro `gravity(400)`



(c) MR-II pro `shaw(400)`



(d) MR-II pro `gravity(400)`

Obrázek 4.5: Srovnání koeficientů z Lanczosova algoritmu pro metody MINRES a MR-II na úlohách `shaw(400)` *s* $RNL = 10^{-7}$ *a* `gravity(400)` *s* $RNL = 10^{-3}$.

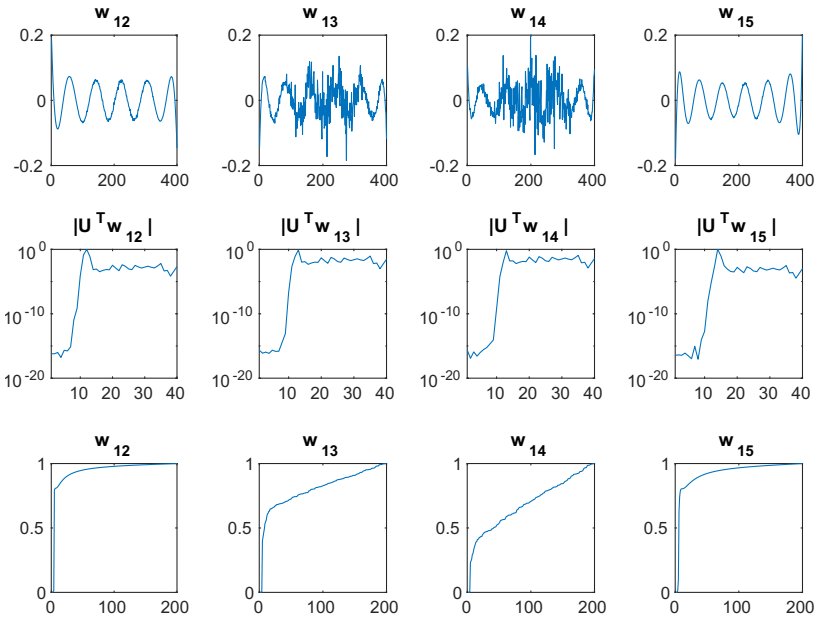
Zdá se, že k tomuto jevu dochází při řešení lineárního problému $T_{j+}y_j \approx \|r_0\|e_1$ v důsledku toho, že koeficienty $\alpha_{j_{rev}}$ a $\beta_{j_{rev}}$ jsou v absolutní hodnotě výrazně nižší než stejné koeficienty pro jiná j. Pro nalezení co nejmenšího rezidua proto musíme řádky matice T_{j+}, které je obsahují, přenásobovat většími hodnotami než ostatní řádky. Detailnější analýza tohoto procesu by byla možná pomocí postupného rozepsání Givensových rotací v jednotlivých iteracích, v této práci se do ní však nebudeme pouštět.

4.3 Porovnání metod LSQR, MINRES a MR-II

Lanczosův algoritmus pro MR-II a MINRES

V Sekci 3.3 jsme vedle metody MINRES představili metodu MR-II, která pracuje s Krylovovým prostorem $\mathcal{K}_j(A,Ab)$, do bázových vektorů tohoto Krylovova prostoru proto není vneseno takové množství šumu. Porovnáme nyní vývoj koeficientu $\varphi_j(0)$ a bázových vektorů při generování Krylovových prostorů $\mathcal{K}_j(A,b)$ a $\mathcal{K}_j(A,Ab)$.

Obrázek 4.5 ukazuje vývoj koeficientů $|\varphi_j(0)|, |\alpha_j|$ a β_j pro úlohy z Kapitoly



Obrázek 4.6: Vektory w_j z úlohy `shaw(400)` *pro $RNL = 10^{-7}$ při použití metody MR-II (tj. se startovacím vektorem $w_1 = Ab/\|Ab\|$), jejich spektra a jejich normalizované kumulativní periodogramy.*

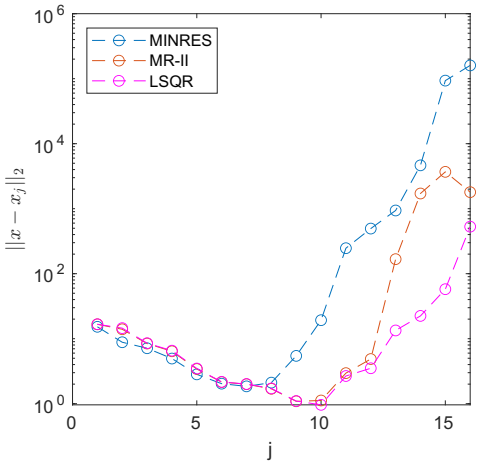
3, jen s tím rozdílem, že začínáme vektorem Ab namísto b. Vidíme, že v obou případech roste koeficient $|\varphi_j(0)|$ výrazně rychleji a k jeho stagnaci dochází výrazně později. V úloze `gravity`, jejíž matice potlačuje vysoké frekvence relativně důkladně, v takovém případě dokonce ani nedochází k poklesu $|\varphi_j(0)|$. Po dosažení iterace $j = 13$ koeficient opět roste. Toto chování je velmi podobné chování vektorů w_j pro výrazně nižší hladinu šumu.

Z Obrázku 4.6 pak vidíme, že význam $\varphi_j(0)$ pro propagaci šumu trvá – v iteraci $j = 13$ přestává $|\varphi_j(0)|$ růst a v téže iteraci se do vektoru w_{13} dostává ve velkém šum. Ze spektra i z periodogramu však vidíme, že i v této iteraci zůstává ve w_{13} určitá dominance hladkých komponent.

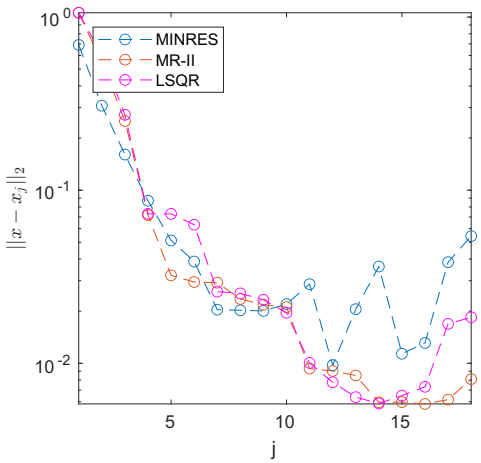
Srovnání normy chyby v různých metodách

V předchozích sekcích jsme se v detailu podívali na koeficient $\varphi_j(0)$ v různých metodách. Pro vzájemné porovnání jednotlivých regularizačních metod však tento koeficient není příliš vhodný – pro Golub-Kahanovu bidiagonalizaci a Lanczosův algoritmus má $\varphi_j(0)$ zcela odlišnou podobu a také dopad na řešení x_j: zatímco v Golub-Kahanově bidiagonalizaci ovlivňuje $\varphi_j(0)$ propagaci šumu pouze do vektorů w_j, ale řešení vytváříme pomocí vektorů s_j, do nichž se nepřesnost teprve musí přenést, v Lanczosově algoritmu se řešení konstruuje přímo pomocí šumem zanesené matice W_j.

V této části proto porovnáme metody LSQR, MINRES a MR-II pro dvě testovací úlohy, konkrétně pro úlohu `phillips(2024)` s $RNL = 10^{-4}$ a pro úlohu `gravity(2024,2,0,1,0.52)` s $RNL = 10^{-5}$. Tato úloha je modifikací problému `gravity(400)`, jen jeho řešení není hladké a singulární hodnoty matice A klesají rychleji k nule. Abychom mohli srovnávat chybu metod pro různé úlohy, definu-

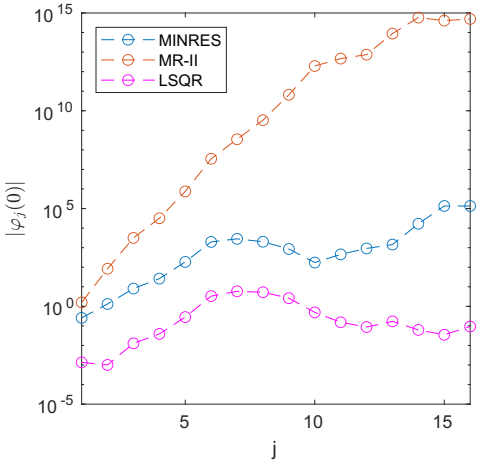


(a) `gravity`

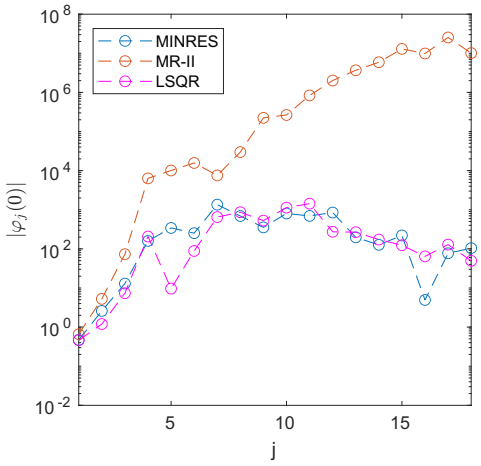


(b) `phillips`

Obrázek 4.7: Srovnání vývoje normy chyby metod MINRES, MR-II a LSQR.

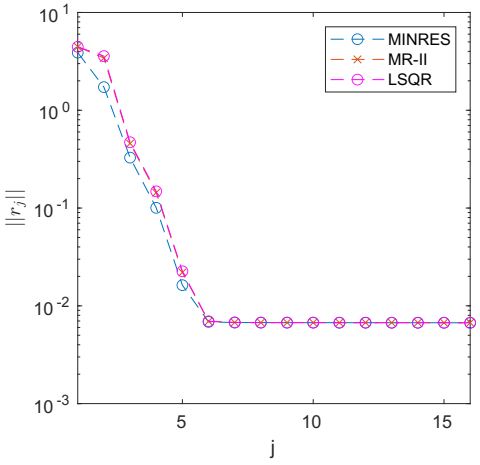


(a) `gravity`

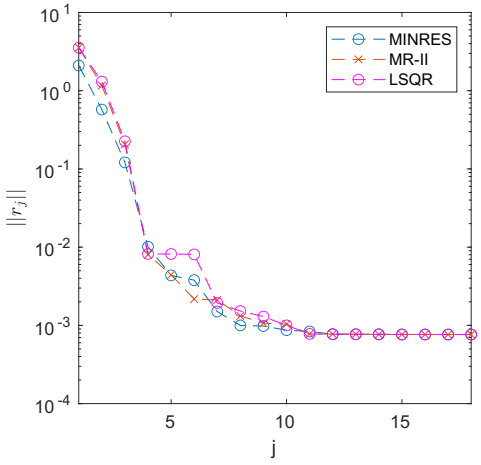


(b) `phillips`

Obrázek 4.8: Srovnání koeficientů $|\varphi_j(0)|$ metod MINRES, MR-II a LSQR.



(a) `gravity`



(b) `phillips`

Obrázek 4.9: Srovnání vývoje norem reziduí metod MINRES, MR-II a LSQR.

	minimální err_{rel}	j_{min}	$\lvert\varphi_{j_{rev}}(0)\rvert$	j_{rev}	$r_{j_{rev}}$
MINRES	0.03099	7	$2.827 \cdot 10^{3}$	7	$8.112 \cdot 10^{-3}$
MR-II	0.01827	9	$1.943 \cdot 10^{12}$	10	$6.820 \cdot 10^{-3}$
LSQR	0.01610	10	5.857	7	$9.123 \cdot 10^{-3}$

Tabulka 4.1: Minimum relativní skutečné chyby, koeficient $\varphi_{j_{rev}}(0)$ a norma rezidua $r_{j_{rev}}$ pro modelovou úlohu `gravity(2024,2,0,1)` *s $RNL = 10^{-5}$. Jako j_{min} označujeme iteraci s nejmenší normou chyby, vzhledem k postupné propagaci šumu do řešení nemusí splývat s j_{rev}.*

	minimální err_{rel}	j_{min}	$\lvert\varphi_{j_{rev}}(0)\rvert$	j_{rev}	$r_{j_{rev}}$
MINRES	$6.766 \cdot 10^{-3}$	9	$1.208 \cdot 10^{3}$	8	$9.987 \cdot 10^{-3}$
MR-II	$1.834 \cdot 10^{-3}$	15	$2.618 \cdot 10^{7}$	16	$7.644 \cdot 10^{-3}$
LSQR	$1.749 \cdot 10^{-3}$	14	$1.306 \cdot 10^{3}$	11	$7.781 \cdot 10^{-3}$

Tabulka 4.2: Minimum relativní skutečné chyby, koeficient $\varphi_{j_{rev}}(0)$ a norma rezidua $r_{j_{rev}}$ pro modelovou úlohu `phillips(2024)` *s $RNL = 10^{-4}$.*

jeme relativní skutečnou chybu aproximace x_k jako

$$err_{rel} = \frac{\|x^{exact} - x_k\|}{\|x_k\|},$$

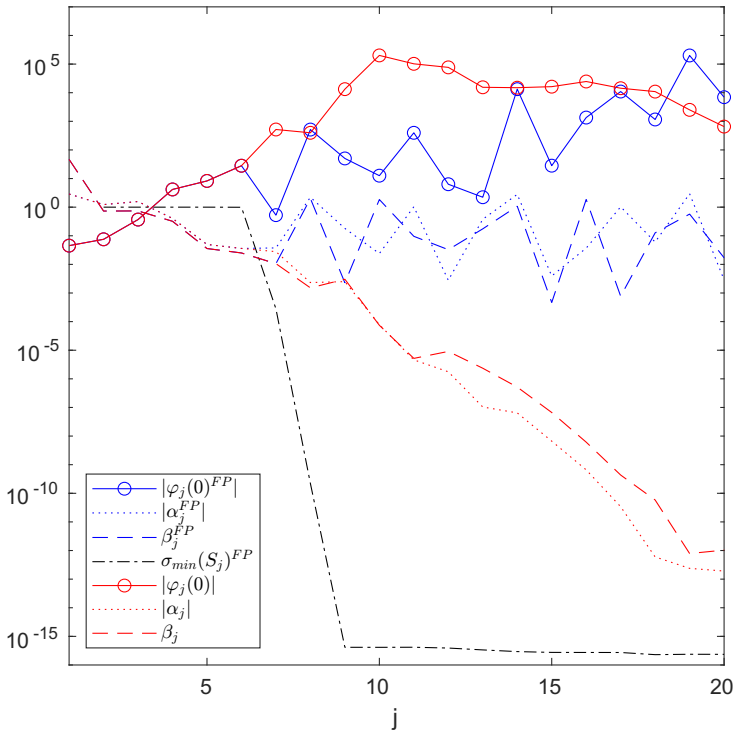
kde x^{exact} je řešení problému $Ax \approx b$ nezatíženého šumem.

Srovnání normy chyby, absolutní hodnoty $\varphi_j(0)$ a norem reziduí jednotlivých metod poskytují Obrázky 4.7, 4.8 a 4.9. Optimální hodnoty co do minimalizace relativní skutečné chyby pak prezentují Tabulky 4.1 a 4.2.

Z Obrázku 4.7 vidíme, že v obou úlohách vykazují všechny tři metody semikonvergenci, přičemž minimální chyby je v metodách MINRES a MR-II dosaženo spolu s maximem $|\varphi_j(0)|$, v metodě LSQR je potřeba ještě několik iterací k propagaci šumu i do vektorů w_j – vidíme, že v obou úlohách to byly tři iterace. Minimální err_{rel} se mezi metodami sice odlišuje, ale řádově zůstává stejná, nejblíže skutečnému řešení se v obou úlohách dostala metoda LSQR. To není příliš překvapivé vzhledem k tomu, že v ní jsou generovány hned dvě ortonormální posloupnosti a vektor x_k je podobně jako v MR-II konstruován v Krylovově prostoru, jehož všechny vektory jsou přenásobené maticí A, a jsou proto zhlazeny. Úskalím této metody zůstává větší časová složitost daná potřebou dvou maticových násobení v každé iteraci.

Srovnávat absolutní hodnoty $|\varphi_j(0)|$ nedává vzhledem k různému matematickému významu těchto koeficientů v MINRES a LSQR dobrý smysl. Je však zajímavé si všimnout, že pro MR-II je $|\varphi_j(0)|$ mnohem větší než pro MINRES, zároveň ani při dosažení j_{rev} tento koeficient nemusí klesat – na Obrázku 4.8 přesně toto pozorujeme pro úlohu `gravity`.

Všechny tři metody se shodují v tom, že v každé iteraci minimalizují normu rezidua, liší se jen množina, v níž toto minimální reziduum hledají. V důsledku toho však vychází norma rezidua v iteraci j_{rev} velmi podobná a norma rezidua, na které se všechny tři metody ustalují, pak vychází téměř stejná – pro úlohu `gravity` zhruba $6.76 \cdot 10^{-3}$, pro úlohu `phillips` zhruba $7.58 \cdot 10^{-3}$. Konkrétní norma rezidua závisí na úloze, kterou chceme řešit, a na RNL šumu v ní, ale všechny tři metody zkonvergují k reziduím s podobnou normou.



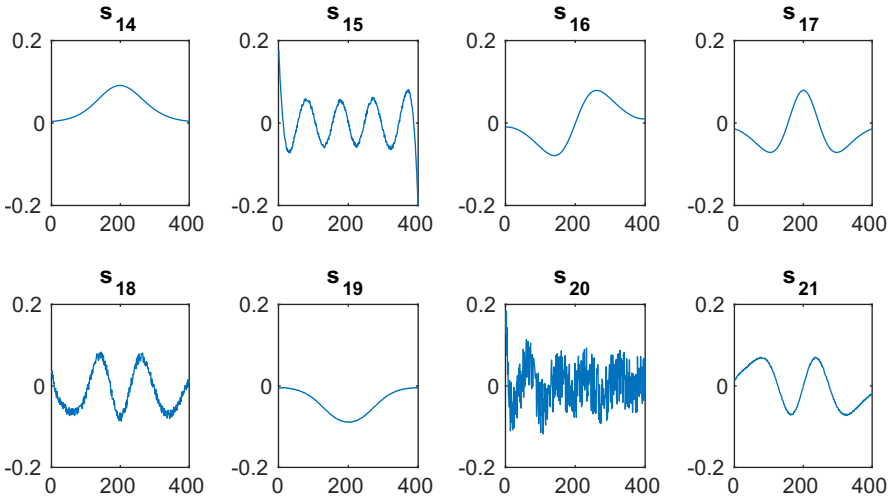
Obrázek 4.10: Koeficienty $|\varphi_j(0)|, |\alpha_j|$ a β_j pro Golub-Kahanovu iterativní bidiagonalizaci pro úlohu `shaw(400)` *s pravou stranou b zatíženou šumem o $RNL = 10^{-7}$. Koeficienty označené indexem FP jsou počítány bez reortogonalizace, koeficienty bez indexu jsou počítány s reortogonalizací, tj. v simulaci přesné aritmetiky. Koeficient $|\varphi_j(0)|$ poměrně silně osciluje, proto v této úloze není vhodné jeho použití coby zastavovacího kritéria – alternativní kritéria navrhují např. [HPS09].*

4.4 Vliv konečné aritmetiky na propagaci šumu

Až dosud jsme v této práci uvažovali veškeré výpočty v přesné aritmetice – v numerických experimentech jsme ji napodobovali alespoň použitím reortogonalizace v algoritmech, které by jinak měly využívat krátké rekurence. V této sekci nastíníme vliv konečné aritmetiky a z ní plynoucí ztráty ortogonality na procesy popsané v této práci. Chování Lanczosova algoritmu v konečné aritmetice studuje [Pai71] nebo [MS06], pro Golub-Kahanovu iterační bidiagonalizaci vyjdeme z [HPS09, 5].

Klíčovým poznatkem pro tuto analýzu je známá tendence, kterou pro čtvercové matice A pozoruje např. [LS12]: Lanczosův algoritmus se v konečné aritmetice často chová tak, že aproximuje větší vlastní čísla matice A ne jedním, ale hned několika blízkými Ritzovými čísly. Důsledkem tohoto jevu je citelné zpomalení konvergence dané numerické metody při řešení klasických soustav lineárních algebraických rovnic. V našem případě, kdy řešíme inverzní úlohu s pravou stranou zanesenou šumem, bychom měli pozorovat zpomalení semikonvergence metody. Zároveň se mezi vektory s_j, resp. w_j opakovaně objevují i jejich hladké komponenty. Dopady tohoto jevu na Golub-Kahanovu bidiagonalizaci ilustruje Obrázek 4.10.

Čerchovaná čára ukazuje nejmenší singulární číslo matice S_j, která by v přesné aritmetice měla být ortogonální a všechna její singulární čísla by měla být rovna



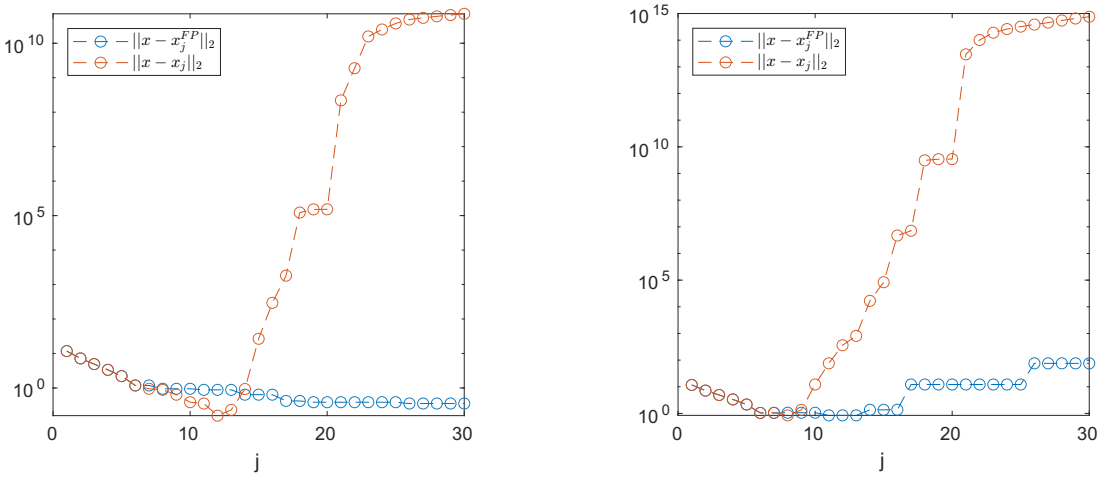
Obrázek 4.11: Vektory s_j pro úlohu `shaw(400)` *s $RNL = 10^{-7}$ spočtené Golub-Kahanovou iterační bidiagonalizací bez reortogonalizace.*

jedné. V konečné aritmetice se však ortogonalita ztrácí a od $j = 9$ je tato matice dokonce numericky singulární. V okamžiku ztráty ortogonality vektorů s_j koeficienty $|\alpha_j(0)|$ a β_j přestávají klesat a jen oscilují, v důsledku toho se zpomaluje nárůst koeficientu $|\varphi_j(0)|$. Zajímavé je, že koeficienty $|\varphi_j(0)|$ v konečné aritmetice následují se zpožděním hodnoty koeficientů v přesné aritmetice – na Obrázku 4.10 pozorujeme, že $|\varphi_8(0)^{FP}| \approx |\varphi_7(0)|$, totéž platí pro $|\varphi_{11}(0)^{FP}|, |\varphi_{14}(0)^{FP}|$ a $|\varphi_{19}(0)^{FP}|$, které odpovídají popořadě $|\varphi_8(0)|, |\varphi_9(0)|$ a $|\varphi_{10}(0)|$.

Opakované zanášení hladkých vektorů do posloupnosti s_j pak ilustruje Obrázek 4.11. Z tohoto obrázku se zdá, že v konečné aritmetice se j_{rev} posunula na $j = 19$. To je v souladu s tím, že koeficient $|\varphi_{19}(0)^{FP}|$ odpovídá $|\varphi_{10}(0)|$, tedy maximálnímu koeficientu v přesné aritmetice. Obrázek 4.12 pak ukazuje vývoj normy chyby v přesné aritmetice – zpoždění, které vidíme u $|\varphi_j(0)|$, můžeme pozorovat i zde. Od chvíle, kdy dojde ke ztrátě ortogonality, se začnou normy chyby před každou změnou několikrát opakovat, odpovídají však normách chyby v přesné aritmetice - např. $\|x^{exact} - x_{26}^{FP}\| \approx \cdots \approx \|x^{exact} - x_{30}^{FP}\| \approx \|x^{exact} - x_9\|$. Jak je vidět z části (b), i v konečné aritmetice trvá semikonvergence této metody. Oscilace $\varphi_j(0)$ a pomalejší konvergence však mohou představovat problém pro některé metody strojové detekce j_{rev}, prostý požadavek na pokles nebo stagnaci $|\varphi_j(0)|, |\alpha_j|$ nebo β_j totiž nemusí stačit.

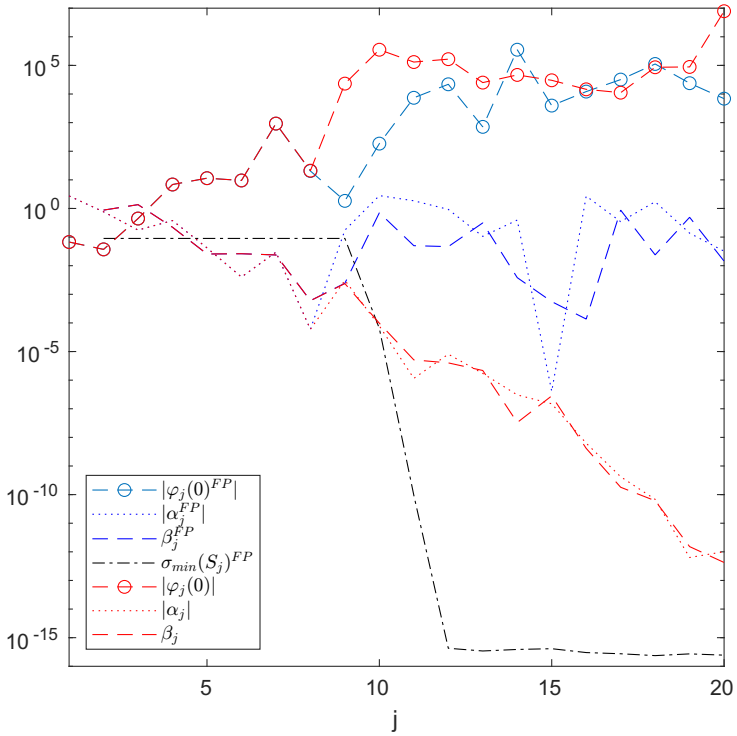
Na Obrázku 4.13 vidíme analogickou analýzu jako na Obrázku 4.10 pro koeficienty z Lanczosova algoritmu. Oba obrázky nemůžeme jednoduše porovnat vzhledem k odlišnému významu koeficientů. I v tomto případě se však po projevení ztráty ortogonality koeficienty začínají rozcházet, $|\alpha_j|$ a β_j přestávají klesat a začínají oscilovat. Ztráta ortogonality přichází v tomto případě o něco později, pravděpodobně v důsledku toho, že v každé iteraci provádíme jen jednu ortogonalizaci namísto dvou. Oproti Golub-Kahanově bidiagonalizaci v tomto případě nepozorujeme výraznější blízkost $|\varphi_j(0)^{FP}|$ a $|\varphi_i(0)|$ pro nějaké $i \leq j$, koeficienty si ale odpovídají alespoň řádově.

Obrázek 4.14 ukazuje, jak se bez reortogonalizace chová norma chyby $\|x^{exact} - x_j\|$ pro aproximaci x_j získanou metodou MINRES. Pozorujeme zde stejné chování jako u metody LSQR – oproti výpočtům s reortogonalizací se zde od ztráty

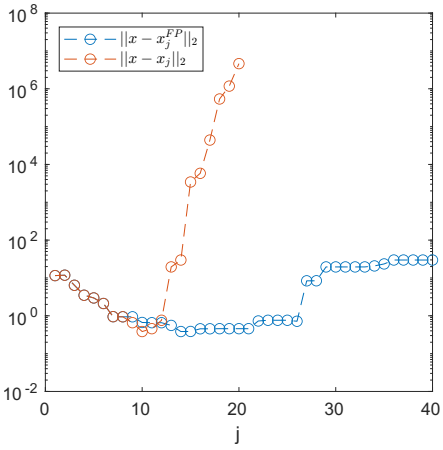


(a) $RNL = 10^{-7}$ *(b)* $RNL = 10^{-3}$

Obrázek 4.12: Norma chyby řešení při výpočtu x_j pomocí metody LSQR pro problém `shaw(400)` *se dvěma různými hladinami šumu. Pro obě hladiny vidíme zpoždění norem chyby v konečné aritmetice oproti aritmetice přesné, pro $RNL = 10^{-3}$ i opožděné dosažení optimálního řešení.*



Obrázek 4.13: Koeficienty $|\varphi_j(0)|, |\alpha_j|$ a β_j pro Lanczosův algoritmus použitý na úlohu `shaw(400)` *s $RNL = 10^{-7}$. Koeficienty označené indexem FP jsou počítány bez reortogonalizace, koeficienty bez indexu jsou počítány s reortogonalizací, tj. v simulaci přesné aritmetiky.*



Obrázek 4.14: Srovnání vývoje normy chyby $\|x^{exact} - x_j\|$ pro x_j počítané metodou MINRES pro testovací problém `shaw(400)` *s $RNL = 10^{-7}$.*

ortogonality začínají normy chyby v každé iteraci několikrát opakovat, i zde však odpovídají normám chyby v přesné aritmetice – optima je nabýváno pro $\|x^{exact} - x_{10}\| \approx \|x^{exact} - x_{13}^{FP}\|$, následně pak např. $\|x^{exact} - x_{12}\| \approx \|x^{exact} - x_{22}^{FP}\| \approx \cdots \approx \|x^{exact} - x_{25}\|$ apod. Také metoda MINRES si tedy v konečné aritmetice se ztrátou ortogonality zachovává semikonvergenci.

Závěr

V diplomové práci jsme se věnovali projekčním metodám řešícím diskrétní inverzní úlohy. První kapitola shrnuje nezbytnou teorii, ukazuje, že diskrétní inverzní problémy, které v práci studujeme, získáme diskretizací Fredholmovy integrální rovnice 1. druhu (1.1), a analyzuje singulární rozvoj matice A, již touto diskretizací získáme. Ukazuje se, že kovariance x^{naive} je závislá na nejmenším singulárním čísle matice A, což v kombinaci s tím, že A je špatně podmíněná, vede k nutnosti použití regularizačních metod. V závěru první kapitoly pak vymezujeme pojem Krylovova prostoru, představujeme obecný princip krylovovských projekčních metod a vysvětlujeme, proč mají tyto metody regularizační vlastnosti.

Ve druhé kapitole je vyložen algoritmus Golub-Kahanovy iterační bidiagonalizace a krylovovská iterativní metoda LSQR, která jej využívá. Následně je shrnuta a podrobněji dokázána teorie z [HPS09] a [HKP17]. Ve Větě 7 tak získáváme explicitní vyjádření pro koeficient $\varphi_j(0)$, jemuž je přímo úměrné množství šumu ve vektorech s_j generovaných Golub-Kahanovou bidiagonalizací. Vliv tohoto koeficientu na bázové vektory s_j a na skutečnou chybu metody LSQR je ilustrován numerickými experimenty. Po vzoru [HKP17] následně dokazujeme Větu 10, která poskytuje vyjádření rezidua r_j^{LSQR} pomocí vektorů s_j a koeficientů $\varphi_j(0)$.

Ve třetí kapitole je představen Arnoldiho a Lanczosův algoritmus a vyložen základní princip metody MINRES a z ní odvozených metod MR-II a GMRES. Pro Lanczosův algoritmus je pak analyzováno šíření šumu ve vektorech w_j a podobně jako v Kapitole 2 je pro koeficient $\varphi_j(0)$, jímž je v těchto vektorech amplifikován šum, nalezen rekurentní i explicitní vztah (zformulovaný ve Větě 14). Ve Větě 16 je pak zformulován vztah mezi normou rezidua r_j^{MINRES} a koeficienty $\varphi_j(0)$. Dopad koeficientu $\varphi_j(0)$ na vektory w_j, na skutečnou chybu a na normu rezidua je v celé kapitole ilustrován numerickými experimenty. Krátce je též zmíněno šíření šumu v Arnoldiho algoritmu, z nějž vychází metoda GMRES, detailnější analýza tohoto jevu však mimo rozsah a možnosti této diplomové práce. Otevřeným problémem, který v této kapitole rovněž není vyřešen, je možné využití poznatků o koeficientu $\varphi_j(0)$ k návrhu zastavovacích kritérií pro metodu MINRES a metody z ní odvozené.

Čtvrtá kapitola obsahuje čtyři numerické experimenty, studující fenomény související s látkou Kapitol 2 a 3. Konkrétně jsou studovány nepravidelnosti propagace šumu v Lanczosově algoritmu a jejich souvislost s chováním koeficientů α_j a β_j z tohoto algoritmu, chování vnitřního řešení y_j z metody MINRES, srovnání metod LSQR, MINRES a MR-II pro stejnou testovací úlohu. Poslední experiment ukazuje propagaci šumu v Golub-Kahanově bidiagonalizaci a v Lanczosově algoritmu v aritmetice s konečnou přesností. Pro všechny čtyři tyto jevy autorovi není známo rigorózní matematické vysvětlení, což skýtá prostor pro další výzkum v této oblasti.

Seznam použité literatury

[Bak77] C. T. H. Baker. *The Numerical Treatment of Integral Equations*. Monographs on numerical analysis. Clarendon Press, 1977.

[BES98] Å. Björck, T. Elfving, and Z. Strakoš. Stability of conjugate gradient and lanczos methods for linear least squares problems. *SIAM Journal on Matrix Analysis and Applications*, 19(3):720–736, 1998.

[BS17] L. Barto and D. Stanovský. *Počítačová algebra*. Matfyzpress, 2017.

[BT18] L. Barto and J. Tůma. *Lineární algebra*. 2018. dostupné online z `http://www.karlin.mff.cuni.cz/~tuma/LA2-17/skripta_la6.pdf`.

[CLR00] D. Calvetti, B. Lewis, and L. Reichel. GMRES-type methods for inconsistent systems. *Linear Algebra and its Applications*, 316(1):157–169, 2000. Special Issue: Conference celebrating the 60th birthday of Robert J. Plemmons.

[CNO07] J. Chung, J. G. Nagy, and D. O'leary. A weighted-gcv method for lanczos-hybrid regularization. *Electronic Transactions on Numerical Analysis*, 28:149–167, 01 2007.

[Cra55] E. J. Craig. The n-step iteration procedures. *Journal of Mathematics and Physics*, 34(1-4):64–73, 1955.

[DM85] L. M. Delves and J. L. Mohamed. *Computational Methods for Integral Equations*. Cambridge University Press, 1985.

[DTHPS12] E. J. Duintjer Tebbens, I. Hnětynková, M. Plešinger, and Z. Strakoš. *Analýza metod pro maticové výpočty: základní metody*. Matfyzpress, 2012.

[GK65] G. Golub and W. Kahan. Calculating the singular values and pseudo-inverse of a matrix. *Journal of the Society for Industrial and Applied Mathematics: Series B, Numerical Analysis*, 2(2):205–224, 1965.

[Gol76] R. R. Goldberg. *Methods of Real Analysis*. Wiley, 1976.

[Had02] J. Hadamard. Sur les problèmes aux dérivées partielles et leur signification physique. *Princeton University Bulletin*, XIII(4):49–52, 1902.

[Han71] R. J. Hanson. A numerical method for solving fredholm integral equations of the first kind using singular values. *SIAM Journal on Numerical Analysis*, 8(3):616–622, 1971.

[Han88] P. C. Hansen. Computation of singlular value expansion. *Computing*, 40:185–199, 1988.

[Han90] P. C. Hansen. The discrete Picard condition for discrete ill-posed problems. *BIT Numerical Mathematics*, 30(4):658–672, 1990.

[Han95] M. Hanke. *Conjugate Gradient Type Methods for Ill-Posed Problems*. Chapman & Hall/CRC Research Notes in Mathematics Series. Taylor & Francis, 1995.

[Han07] P. C. Hansen. Regularization tools version 4.0 for MATLAB 7.3. *Numerical Algorithms*, 46:189–194, 11 2007.

[Han10] P. C. Hansen. *Discrete Inverse Problems*. Society for Industrial and Applied Mathematics, 2010.

[Han17] M. Hanke. *A Taste of Inverse Problems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2017.

[HJ85] Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, 1985.

[HKP17] I. Hnětynková, M. Kubínová, and M. Plešinger. Noise representation in residuals of LSQR, LSMR, and CRAIG regularization. *Linear Algebra and its Applications*, 533:357–379, 2017.

[HPS09] I. Hnětynková, M. Plešinger, and Z. Strakoš. The regularizing effect of the Golub-Kahan iterative bidiagonalization and revealing the noise level in the data. *BIT Numerical Mathematics*, 49:669–696, 2009.

[HS52] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of research of the National Bureau of Standards*, 49:409–436, 1952.

[JH07] T.K. Jensen and P.C. Hansen. Iterative regularization with minimum-residual methods. *BIT Numerical Mathematics*, 47:103–120, 2007.

[Kre99] R. Kress. *Linear Integral Equations*. Applied Mathematical Sciences. Springer New York, 1999.

[Lan50] C. Lanczos. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *Journal of research of the National Bureau of Standards*, 45:255–282, 1950.

[LS12] J. Liesen and Z. Strakoš. *Krylov Subspace Methods: Principles and Analysis*. Oxford University Press, 2012.

[Mic13] M. Michenková. Regularizační metody založené na metodách nejmenších čtverců. Master's thesis, Univerzita Karlova, 2013.

[MN02] J. Matoušek and J. Nešetřil. *Kapitoly z diskrétní matematiky*. Univerzita Karlova v Praze, Nakladatelství Karolinum, 2002.

[MS06] G. Meurant and Z. Strakoš. The Lanczos and conjugate gradient algorithms in finite precision arithmetic. *Acta Numerica*, 15:471–542, 2006.

[Pai71] C. C. Paige. *The computation of eigenvalues and eigenvectors of very large sparse matrices*. PhD thesis, University of London, 1971.

[Pra78] W.K. Pratt. *Digital Image Processing*. Number sv. 1 in A Wiley-Interscience publication. Wiley, 1978.

[PS75] C. C. Paige and M. A. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM Journal on Numerical Analysis*, 12(4):617–629, 1975.

[PS82] C. C. Paige and M. A. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Trans. Math. Softw.*, 8(1):43–71, 1982.

[Saa03] Y. Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, second edition, 2003.

[SS86] Y. Saad and M. H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 7(3):856–869, 1986.

[Tik63] A. N. Tikhonov. Solution of incorrectly formulated problems and the regularization method. *Soviet Math. Dokl.*, 4:1035–1038, 1963.

[Vas11] K. Vasilík. Lineární algebraické modelování úloh s nepřesnými daty. Master's thesis, Univerzita Karlova, 2011.

[Vog02] C. R. Vogel. *Computational Methods for Inverse Problems*. Society for Industrial and Applied Mathematics, 2002.