Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra matematiky

Bakalářská práce

Metody řešení normálních rovnic při vyrovnání geodetických sítí

Plzeň, 2006 Hana Kutáková

Prohlášení

Předkládám tímto k posouzení a obhajobě bakalářskou práci zpracovanou na závěr studia na Fakultě aplikovaných věd Západočeské univerzity v Plzni.

Prohlašuji, že jsem předloženou bakalářskou práci vypracovala samostatně s použitím odborné literatury a pramenů, jejichž úplný seznam je součástí práce, a za odborného vedení vedoucího bakalářské práce.

V Plzni dne 12. června 2006 .....

podpis

Poděkování

Ráda bych touto cestou poděkovala především panu Ing. Vratislavu Fillerovi, Ph.D. za ochotu, s jakou přistupoval k vedení této práce, za rady a čas, které mi věnoval na konzultacích.

Dále bych chtěla poděkovat panu Ing. Janu Frischovi za to, že mě přivedl k tématu této práce, a za doporučení řady cenných materiálů. Zároveň děkuji Ing. Martinu Kadlecovi za poskytnutí části testovacích dat pro tuto práci.

Abstrakt

Součástí geodetické praxe je výpočet a vyrovnání naměřených hodnot. Ve vyrovnání vystupují rozsáhlé soustavy lineárních algebraických rovnic, tzv. normální rovnice. Tato práce se zabývá jejich řešením. Je zaměřena na iterační metody, především na metody gradientního typu. Klasické algoritmy jsou následně modifikovány s důrazem na využití vlastností matice normálních rovnic. Jeden z použitých způsobů modifikace je metoda předpodmínění. Podstatnou částí práce je vlastní implementace zkoumaných metod, která je realizována v programu Matlab®. Testování metod je provedeno na reálných datech z geodetické praxe. V závěru práce jsou dosažené výsledky vhodným způsobem porovnány a zhodnoceny.

Klíčová slova

Normální rovnice, soustavy rovnic, iterační metody, symetrická a pozitivně definitní matice, gradientní metody, singulární rozklad matice, předpodmínění, neúplný Choleského rozklad, residuum, Matlab®

Abstract

Calculation of measured values is important part of geodetic practice. In the adjustment rises large systems of linear equations, so-called normal equations. This work deals with methods of their solving. The work is focused on iterative methods, especially on gradient methods and their modification, which profit from special structure of normal equation matrix. One of the applied modification method is preconditioning. The major part of this work deals with implementation of considered methods, programmed in Matlab®. All methods are tested on normal equation system created from real measurements. Finally, all considered methods are appropriately compared and evaluated from more points of view.

Keywords

Normal equation, system of linear equations, iterative methods, symmetric positive definite matrix, gradient methods, singular value decomposition, preconditioning, incomplete Cholesky factorization, residual, Matlab®

Obsah

Kapitola 1

Úvod

Po provedení geodetických měření v terénu následuje výpočet a vyrovnání naměřených hodnot, nejčastěji metodou nejmenších čtverců. Zpravidla se setkáváme s rozsáhlými soubory dat, pro jejichž zpracování je potřeba hledat vhodné postupy umožňující efektivní řešení. Použití metody nejmenších čtverců vede k řešení soustavy lineárních rovnic se specifickými vlastnostmi. V některých případech je potřeba brát je v úvahu a s ohledem na ně volit odpovídající programové řešení. Lze tak docílit zlepšení efektivity výpočtu a v případě iteračních postupů zajistit větší rychlost konvergence k požadovanému řešení.

1.1 Cíle práce

V této práci se zaměříme na problematiku řešení normálních rovnic, které vznikají například při vyrovnání plošných geodetických sítí. Bude nás zajímat pohled z hlediska numerické matematiky, tedy metody řešení rozsáhlých soustav lineárních algebraických rovnic. Zaměříme se na iterační metody. Seznámíme se s postupy, které se v praxi běžně používají, nejvíce nás budou zajímat metody gradientního typu. Poté se pokusíme tyto metody různými způsoby modifikovat, využijeme vlastnosti matice normálních rovnic. Budeme tak chtít dosáhnout lepších numerických vlastností samotných výpočtů, po-

kusíme se zmenšit počet iterací potřebných k získání řešení s požadovanou přesností a zkrátit dobu výpočetního cyklu.

Podstatnou součástí této práce bude vlastní implementace metod. Programové řešení budeme realizovat v programu Matlab®. Vyvinuté metody bude potřeba vhodným způsobem zhodnotit. Pro simulaci reálné situace z geodetické praxe provedeme testování metod na reálných datech. Abychom alespoň částečně zajistili větší objektivitu, budeme pracovat se dvěma soustavami normálních rovnic.

Na základě testovacích výpočtů provedeme zhodnocení získaných výsledků. Porovnáme výsledky dosažené pomocí klasických iteračních metod s výsledky získanými modifikovanými metodami. Porovnání provedeme několika způsoby, aby hodnocení bylo komplexní.

Praktickým výstupem této práce bude třída funkcí v programu Matlab®, které řeší soustavy normálních rovnic vystupujících při vyrovnání geodetických sítí.

Kapitola 2

Teoretická část – úvod do teorie

2.1 Základní pojmy

V této kapitole definujeme některé základní pojmy. Nebudeme vysvětlovat všechny odborné termíny vyskytující se v této práci, použitá terminologie vychází z pojmů zavedených v předmětu Numerické metody (KMA/NM), více v příslušné literatuře [2].

Horní trojúhelníková matice: Horní trojúhelníková matice je matice $\mathbf{U} = (u_{ij})$, pro kterou platí: $u_{ij} = 0$ pro všechna $i > j$.

Dolní trojúhelníková matice: Dolní trojúhelníková matice je matice $\mathbf{L} = (l_{ij})$, pro kterou platí: $l_{ij} = 0$ pro všechna $i < j$.

Řídká matice Řídké matice představují speciální třídu matic, jejichž převážná většina prvků se rovná nule.

Pozitivně (semi)definitní matice Symetrická matice $\mathbf{A} \in \mathbb{R}^{n \times n}$ se nazývá pozitivně semidefinitní, jestliže $\mathbf{x}^T\mathbf{A}\mathbf{x} \geq 0$ pro každé $\mathbf{x} \in \mathbb{R}^n$, a pozitivně definitní, jestliže $\mathbf{x}^T\mathbf{A}\mathbf{x} > 0$ pro všechna nenulová $\mathbf{x} \in \mathbb{R}^n$.

Ortogonální matice: Matice $\mathbf{Q} \in \mathbb{R}^{n \times n}$ je ortogonální, jestliže $\mathbf{Q}^T\mathbf{Q} = \mathbf{I}$, kde $\mathbf{I}$ je jednotková matice. Tedy platí, že $\mathbf{Q}^T = \mathbf{Q}^{-1}$. Ortogonální matice mají následující vlastnosti:

- Řádky i sloupce tvoří ortonormální vektory, tedy řádky i sloupce matice $\mathbf{Q}$ jsou v normě jednotkové vektory.
- $\mid q_{ij} \mid \leq 1$ a $\mid (q^{-1})_{ij} \mid \leq 1, \quad \forall i, j, \quad$ kde $(q^{-1})_{ij}$ jsou prvky inverzní matice.

Tyto vlastnosti jsou z numerického hlediska velmi důležité, druhá vlastnost zaručuje, že prvky inverzní matice nemohou během výpočtu nekontrolovatelně růst. To je jeden z důvodů, proč mají ortogonální matice rozsáhlé aplikace v numerické lineární algebře [8].

Vektory A–sdružené [1] Řekneme, že vektory $\{\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_n\}$ jsou navzájem sdružené vzhledem k symetrické pozitivně definitní matici $\mathbf{A}$, jestliže platí

$$\mathbf{d}_i^T \mathbf{A} \mathbf{d}_j = 0, \ \forall i \neq j.$$

Číslo podmíněnosti matice Číslo podmíněnosti čtvercové matice $\mathbf{A}$ je definováno

$$\kappa(\mathbf{A}) = \|\mathbf{A}\| \, \|\mathbf{A}^{-1}\|$$

a vyjadřuje míru stability a citlivosti matice $\mathbf{A}$ vzhledem k numerickým operacím. V případě, že matice $\mathbf{A}$ je symetrická a pozitivně definitní, lze podle [6] číslo podmíněnosti definovat jako podíl největšího a nejmenšího vlastního čísla matice $\mathbf{A}$

$$\kappa(\mathbf{A}) = \frac{\lambda_{max}(\mathbf{A})}{\lambda_{min}(\mathbf{A})}.$$

Pro $\kappa(\mathbf{A}) \approx 1$ je matice velmi dobře podmíněná. S rostoucí hodnotou $\kappa(\mathbf{A})$ se zhoršují numerické vlastnosti matice, řádově velké číslo podmíněnosti znamená, že matice se blíží k singulární matici. Soustavy rovnic s maticí s vysokým číslem podmíněnosti jsou obtížně řešitelné.

Výpočetní složitost Výpočetní složitost numerické metody je počet matematických operací sčítání, odčítání, násobení a dělení nutných k získání výsledku (viz [16]).

[1] z anglického A-conjugate

2.2 Normální rovnice

Jedním ze způsobů vyrovnání naměřených dat je vyrovnání zprostředkujících pozorování, kdy hodnoty hledaných veličin určujeme nepřímo, zprostředkovaně pomocí měření jiných veličin, které lze přímo měřit a které jsou v přímém funkčním vztahu s hledanými neznámými veličinami (více viz [1]). Vycházíme z toho, že známe vektor $\mathbf{x}_0$ přibližných hodnot neznámých veličin, v našem případě přibližné souřadnice určovaných bodů. Funkční vztahy zapíšeme

$$\mathbf{F}(\mathbf{x}) = \mathbf{l} + \mathbf{v}, \tag{2.1}$$

kde $\mathbf{l}$ je vektor naměřených zprostředkujících veličin rozměru $n \times 1$, $\mathbf{v}$ je vektor oprav rozměru $n \times 1$. Tento vztah je obecně nelineární, je nutno provést linearizaci

$$\mathbf{F}(\mathbf{x}_0) + \mathbf{A}\mathbf{dx} = \mathbf{l} + \mathbf{v}, \tag{2.2}$$

po úpravě

$$\mathbf{A}\mathbf{dx} + \mathbf{F}(\mathbf{x}_0) - \mathbf{l} = \mathbf{v}, \tag{2.3}$$

kde $\mathbf{A}$ je matice plánu rozměru $n \times k$ tvořená

$$\mathbf{A} = \frac{\partial \mathbf{F}(\mathbf{x})}{\partial \mathbf{x}}.$$

Vektor $\mathbf{dx}$ je vektor hledaných neznámých přírůstků přibližných souřadnic neznámých bodů, je rozměru $k \times 1$. Označíme $\mathbf{L} = \mathbf{F}(\mathbf{x}_0) - \mathbf{l}$ a vztah (2.3) přejde do tvaru

$$\mathbf{A}\mathbf{dx} + \mathbf{L} = \mathbf{v}. \tag{2.4}$$

Dále definujeme matici vah $\mathbf{P}$ rozměru $n \times n$, která má zpravidla tvar diagonální matice

$$\mathbf{P} = \begin{pmatrix} p_1 & 0 & \cdots & 0 \\ 0 & p_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & p_n \end{pmatrix}$$

a platí vztah $\mathbf{P} = \mathbf{Q}^{-1}$, kde $\mathbf{Q}$ je její inverzní matice, tzv. matice váhových koeficientů. Systém rovnic musí splňovat podmínku

$$\mathbf{v}^T\mathbf{P}\mathbf{v} = \min. \tag{2.5}$$

Po úpravách dostáváme soustavu rovnic

$$\mathbf{A}^T\mathbf{P}\mathbf{A}\mathbf{dx} + \mathbf{A}^T\mathbf{P}\mathbf{L} = 0, \tag{2.6}$$

které nazýváme normální rovnice. Řešením je vektor $\mathbf{dx}$, který představuje opravy přibližných souřadnic neznámých bodů. Pro další práci se soustavou normálních rovnic označíme

$$\mathbf{A}^T\mathbf{P}\mathbf{A} = \mathbf{N} \tag{2.7}$$

$$\mathbf{A}^T\mathbf{P}\mathbf{L} = -\mathbf{b}. \tag{2.8}$$

Soustava lineárních algebraických rovnic (2.6) pak bude mít tvar

$$\mathbf{N}\mathbf{dx} = \mathbf{b}. \tag{2.9}$$

Matice $\mathbf{N}$ má tyto vlastnosti:

- je symetrická
- je pozitivně definitní
- má řídkou strukturu

Řídkost matice N plyne z toho, jakým způsobem matice vznikla (viz (2.7)). Matice $\mathbf{A}$ je řídká a matice $\mathbf{P}$ je zpravidla diagonální, tedy matice $\mathbf{N}$ musí být také řídká.

Velikost (počet řádků, resp. sloupců) matice $\mathbf{N}$ se pohybuje v řádu 10^2 až 10^3 při vyrovnání plošných sítí, 10^4 při zpracování GPS[2] měření. V případě zpracování GPS měření se podle [13] v určitých částech výpočtu řeší celočíselná metoda nejmenších čtverců, což vede ke zvláštním výpočetním postupům.

[2] z anglického Global Positioning System – Globální polohový systém

Kapitola 3

Teoretická část – standardní metody

V následujících kapitolách se budeme zabývat numerickými metodami řešení normálních rovnic. Využijeme výše uvedených vlastností matice normálních rovnic. Abychom se pohybovali ve standardním značení, soustavu normálních rovnic (2.9) formálně nahradíme tvarem

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \tag{3.1}$$

matice $\mathbf{A} \in \mathbb{R}^{n \times n}$ je matice[1] s výše uvedenými vlastnostmi matice $\mathbf{N}$, $\mathbf{x} \in \mathbb{R}^n$ je vektor neznámých a $\mathbf{b} \in \mathbb{R}^n$ je vektor pravé strany.

Numerické metody řešení soustav lineárních algebraických rovnic se dělí na metody přímé a iterační. Velice stručně se zmíníme o přímých metodách a zaměříme se na metody iterační.

3.1 Přímé metody

Jedná se o metody, která vedou k přesnému řešení (nebereme-li v úvahu vliv zaokrouhlovacích chyb) po konečném počtu kroků (viz [2]). Mezi přímé me-

[1] Tato matice nemá nic společného s maticí $\mathbf{A}$ uvedené v podkapitole Normální rovnice, pouze formálně nahrazuje matici $\mathbf{N}$.

tody patří Gaussova eliminační metoda (tu lze různými způsoby modifikovat, mluvíme potom o Gaussově eliminační metodě se sloupcovou, řádkovou nebo úplnou pivotací), metoda LU rozkladu (v případě symetrické a pozitivně definitní matice mluvíme o Choleského rozkladu). Vzhledem k pozdějšímu využití si ukážeme princip Choleského rozkladu.

Pokud chceme využít vlastnosti matice normálních rovnic, jeví se použití většiny přímých metod jako nevhodné, s výjimkou metody Choleského rozkladu (využívá symetrii a pozitivní definitnost matice soustavy). Použití přímých metod (např. Gaussova eliminace) je vhodné pro libovolné plné matice, nevyužívají informaci o struktuře matice (např. řídkost). Pro řídké matice (jako je matice normálních rovnic) může dojít k zaplnění, protože matice soustavy se v průběhu výpočtu mění. Jednou z možností by bylo použít vhodnou přímou metodu a upravit ji tak, abychom využili řídkost matice, například vhodnou pivotací, která by minimalizovala zaplnění. Druhou možností je použít iterační metody místo přímých. Na tento druhý způsob se zaměříme.

Výpočetní složitost (počet matematických operací, které je nutno během výpočtu vykonat) přímých metod se podle [10] pohybuje v řádu $c \cdot n^3$, kde c je konstanta a n je řád matice soustavy. Konkrétně například pro Gaussovu eliminační metodu $\frac{2}{3}n^3$. Jsou vhodné pro řešení soustav nižších řádů.

3.1.1 Choleského rozklad

Uvažujme soustavu (3.1) se symetrickou a pozitivně definitní maticí $\mathbf{A}$. Pro každou pozitivně definitní matici $\mathbf{A}$ lze nelézt jednoznačně určenou dolní trojúhelníkovou matici $\mathbf{G}$ s kladnými diagonálními prvky takovou, že platí

$$\mathbf{A} = \mathbf{G}\mathbf{G}^T. \tag{3.2}$$

Při řešení postupujeme takto:
Provedeme Choleského rozklad $\mathbf{A} = \mathbf{G}\mathbf{G}^T$. Dosazením do rovnice (3.1) za $\mathbf{A}$

dostaneme $\mathbf{Ax} = \mathbf{GG}^T\mathbf{x} = \mathbf{b}$. Označíme $\mathbf{G}^T\mathbf{x} = \mathbf{y}$. Následuje postupné řešení řešení dvou soustav lineárních algebraických rovnic, a to

$$\mathbf{Gy} = \mathbf{b} \tag{3.3}$$

$$\mathbf{G}^T\mathbf{x} = \mathbf{y} \tag{3.4}$$

Z rovnice (3.3) vypočítáme vektor $\mathbf{y}$ a následně z rovnice (3.4) vektor $\mathbf{x}$. Vzhledem k tomu, že $\mathbf{G}$ je dolní trojúhelníková matice (a $\mathbf{G}^T$ tedy horní trojúhelníková matice), lze soustavy vyřešit přímou, resp. zpětnou substitucí. Nalezený vektor $\mathbf{x}$ je pak řešením původní soustavy $\mathbf{Ax} = \mathbf{b}$.

3.2 Iterační metody

Opět se zabýváme řešením soustavy (3.1). Na rozdíl od přímých metod se v jednotlivých iteracích (číslo iterace označíme např. k) k přesnému řešení $\mathbf{x}^*$ pouze přibližujeme, v limitě pak dostáváme

$$\mathbf{x}^* = \lim_{k \to +\infty} \mathbf{x}^{(k)}.$$

Soustavu (3.1) lze přepsat do tvaru

$$\mathbf{x} = \mathbf{Hx} + \mathbf{g}, \tag{3.5}$$

matice $\mathbf{H}$ se nazývá iterační matice. V jednotlivých iteracích má soustava (3.5) tvar buď

$$\mathbf{x}^{(k+1)} = \mathbf{Hx}^{(k)} + \mathbf{g}, \tag{3.6}$$

kdy mluvíme o stacionární iterační metodě (matice $\mathbf{H}$ a vektor $\mathbf{g}$ se během výpočtů nemění), nebo

$$\mathbf{x}^{(k+1)} = \mathbf{H}^{(k)}\mathbf{x}^{(k)} + \mathbf{g}^{(k)}, \tag{3.7}$$

pak se jedná o nestacionární iterační metodu (matice $\mathbf{H}$ a vektor $\mathbf{g}$ se v každém iteračním kroku mění). Kromě iterační formule (3.6), resp. (3.7) musíme

při řešení soustav iteračními metodami ještě definovat volbu počáteční aproximace $\mathbf{x}^{(0)}$ a stanovit zastavovací podmínku, tedy kdy se algoritmus ukončí. V této práci budeme u všech iteračních metod volit počáteční aproximaci $\mathbf{x}^{(0)} = (0, 0, \ldots, 0)$. Realizace zastavovací podmínky je možná více způsoby. Lze například stanovit, kolik iterací se má provést. Nebo můžeme definovat číslo δ , resp. ε a výpočet ukončíme, bude-li platit

$$\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\| < \delta, \quad \text{resp.} \quad \|\mathbf{x}^{*} - \mathbf{x}^{(k)}\| < \varepsilon.$$

Pak dostáváme aproximaci řešení s přesností δ, resp. s chybou ε. Pokud ovšem neznáme hodnotu přesného řešení $\mathbf{x}^{*}$ (jako v našem případě), je tento způsob ukončení výpočtu bezpředmětný. Jinou možností je sledovat v každé iteraci velikost residua

$$\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}.$$

Mezi stacionární iterační metody patří například Jacobiova metoda, Gaussova–Seidelova metoda, modifikace Gaussovy–Seidelovy metody — metoda SOR[2], mezi nestacionární metody například metoda sdružených gradientů.

Mezi iterační metody řadíme i metodu singulárního rozkladu matice (SVD[3]). Klasický singulární rozklad matice je uvažován pro obecnou obdélníkovou matici $\mathbf{A}$, která nemá žádné speciální vlastnosti. Ukážeme si, jak provést singulární rozklad v případě, že matice soustavy je symetrická.

Výpočetní složitost iteračních metod je řádově $c \cdot n^2 \cdot k$, kde c je konstanta, n je řád matice soustavy a k je počet iterací. Aby byla výpočetní složitost iterační metody rovna nebo lepší než složitost přímých metod, musí být počet iterací potřebný pro nalezení řešení roven n nebo menší než n.

V následujících kapitolách popíšeme nejprve právě singulární rozklad matice, potom se budeme věnovat gradientním metodám.

[2] z anglického Successive Over–Relaxation

[3] z anglického Singular Value Decomposition

3.2.1 Singulární rozklad

Mějme dánu symetrickou matici $\mathbf{A} \in \mathbb{R}^{n\times n}$. Potom existuje ortogonální matice $\mathbf{Q} \in \mathbb{R}^{n\times n}$ taková, že platí

$$\mathbf{Q}^T\mathbf{A}\mathbf{Q} = \mathbf{D} = \mathrm{diag}(\lambda_1, \ldots, \lambda_n), \tag{3.8}$$

kde $\lambda_1, \ldots, \lambda_n$ jsou vlastní čísla matice $\mathbf{A}$ a platí, že $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$. Pokud je symetrická matice $\mathbf{A}$ navíc pozitivně definitní, platí dokonce $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n > 0$. Tento rozklad je nazýván symetrický Schurův rozklad (viz [4]), nebo mluvíme o tzv. spektrální větě pro symetrické matice (viz [8]). Vzhledem k vlastnostem ortogonálních matic ($\mathbf{Q}^T = \mathbf{Q}^{-1}$) je zřejmé, že platí

$$\mathbf{Q}\mathbf{D}\mathbf{Q}^T = \mathbf{A}. \tag{3.9}$$

Takto je singulární rozklad definován v případě symetrické matice $\mathbf{A}$. V případě obecné obdélníkové matice bychom v rozkladu dostali dvě různé ortogonální matice místo jedné matice $\mathbf{Q}$, matice $\mathbf{D}$ by na diagonále neměla vlastní čísla matice $\mathbf{A}$, ale tzv. singulární čísla matice $\mathbf{A}$ (proto singulární rozklad). V případě symetrické matice $\mathbf{A}$ jsou singulární a vlastní čísla totožná. Vraťme se k symetrickému rozkladu (3.8). V momentě, kdy známe singulární rozklad matice $\mathbf{A}$, můžeme soustavu (3.1) přepsat do tvaru

$$\mathbf{Q}\mathbf{D}\mathbf{Q}^T\mathbf{x} = \mathbf{b}. \tag{3.10}$$

Po přenásobení rovnice zleva maticí $\mathbf{Q}^T$ dostaneme

$$\mathbf{D}\mathbf{Q}^T\mathbf{x} = \mathbf{Q}^T\mathbf{b} \tag{3.11}$$

a označíme

$$\mathbf{Q}^T\mathbf{x} = \mathbf{z} \tag{3.12}$$

$$\mathbf{Q}^T\mathbf{b} = \mathbf{d}. \tag{3.13}$$

Dostaneme soustavu

$$\mathbf{D}\mathbf{z} = \mathbf{d}. \tag{3.14}$$

Z rovnice (3.14) tedy snadno vypočítáme vektor $\mathbf{z}$ ($\mathbf{D}$ je diagonální matice), ze znalosti vektoru $\mathbf{z}$ pak z rovnice (3.12) vypočítáme hledané řešení $\mathbf{x}$

$$\mathbf{x} = \mathbf{Q}\mathbf{z}. \tag{3.15}$$

Je zřejmé, že v momentě, kdy známe singulární rozklad matice soustavy $\mathbf{A}$, je řešení soustavy triviální. Otázkou tedy je, jak nalézt matice $\mathbf{Q}$ a $\mathbf{D}$. Jeden z možných algoritmů je popsán v [4] v podobě tzv. symetrického QR–algoritmu. Jedná se v podstatě o aplikaci QR–rozkladu na třídiagonální matici. Nejprve provedeme třídiagonální rozklad matice $\mathbf{A}$ ve tvaru $\mathbf{Z}^T\mathbf{A}\mathbf{Z} = \mathbf{T}$, kde $\mathbf{Z}$ je ortogonální matice a $\mathbf{T}$ je třídiagonální matice. Následně hledáme matici $\mathbf{D}$. V první aproximaci položíme $\mathbf{D}^{(1)} = \mathbf{T}$, následuje iterační postup, jehož podstatou je QR–rozklad matice $\mathbf{D}^{(i)}$. QR–rozklad spočívá v rozkladu matice $\mathbf{T}$ na součin ortogonální matice a horní trojúhelníkové matice. Nediagonální prvky matice $\mathbf{D}$ se v postupných iteracích blíží k nule, na diagonále dostáváme aproximace vlastních, tedy singulárních čísel matice $\mathbf{A}$.

Matice $\mathbf{D}$ je rozkladem určena jednoznačně (důkaz lze nalézt např. v [8]), zatímco matice $\mathbf{Q}$ rozkladem jednoznačně určena není. Pokud je totiž (3.9) singulární rozklad matice $\mathbf{A}$, potom pro ortogonální matici $\mathbf{Q}_0 = -\mathbf{Q}$ platí opět $\mathbf{Q}_0\mathbf{D}\mathbf{Q}_0^T = \mathbf{A}$, přičemž $\mathbf{Q}_0 \neq \mathbf{Q}$.

Vzhledem k použití ortogonálních matic patří singulární rozklad matice k velmi stabilním metodám. Odpadá totiž riziko růstu hodnot prvků nad únosnou mez. Výpočetní složitost se pohybuje v řádu n^3, kde n je rozměr matice soustavy.

3.2.2 Gradientní metody

Uvažujeme symetrickou a pozitivně definitní matici $\mathbf{A}$ v soustavě (3.1). Řešení soustavy (3.1) je ekvivalentní s hledáním minima funkcionálu

$$\phi(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} - \mathbf{x}^T\mathbf{b}, \tag{3.16}$$

kde $\mathbf{b} \in \mathbb{R}^n$ a $\mathbf{A} \in \mathbb{R}^{n \times n}$ je pozitivně definitní matice. Minimum funkcionálu $\phi(\mathbf{x})$ je podle pravidel o derivování matic

$$\nabla\phi(\mathbf{x}) = \frac{1}{2}(\mathbf{A}\mathbf{x} + \mathbf{A}^T\mathbf{x}) - \mathbf{b} = \mathbf{A}\mathbf{x} - \mathbf{b} = \mathbf{0}, \tag{3.17}$$

hodnota minima $\phi(\mathbf{x})$ je pak rovna $-\frac{1}{2}\mathbf{b}^T\mathbf{A}^{-1}\mathbf{b}$, a to pro $\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$, což je na první pohled zřejmé z rovnice (3.17). Tím jsme dokázali, že hledání minima $\phi(\mathbf{x})$ a řešení rovnice $\mathbf{A}\mathbf{x} = \mathbf{b}$ je ekvivalentní, samozřejmě v případě, že matice $\mathbf{A}$ je symetrická a pozitivně definitní. Dále stanovíme počáteční aproximaci $\mathbf{x}^{(0)}$ a hledáme vhodný směr, resp. směry podél kterých hledáme minimum funkcionálu $\phi(\mathbf{x})$, tedy řešení rovnice (3.1). Otázkou je, jakým způsobem minimum hledat. Ukážeme si dvě metody, metodu největšího spádu a poté velmi sofistikovanou metodu sdružených gradientů.

Metoda největšího spádu

V každém bodě $\mathbf{x}$ funkcionál $\phi(\mathbf{x})$ klesá nejvíce ve směru opačném ke směru gradientu $-\nabla\phi(\mathbf{x}) = \mathbf{A}\mathbf{x} - \mathbf{b}$, který v příslušné k-té iteraci označíme

$$\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(k)} \tag{3.18}$$

a nazveme residuem. Každá další aproximace $\mathbf{x}$ je pak dána vztahem

$$\mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} + \alpha_k\mathbf{r}^{(k-1)}, \tag{3.19}$$

kde α_k je hodnota, která udává délku kroku ve směru $\mathbf{r}^{(k-1)}$. Platí, že $\phi(\mathbf{x}^{(k-1)} + \alpha_k\mathbf{r}^{(k-1)}) < \phi(\mathbf{x}^{(k-1)})$, resp. $\phi(\mathbf{x} + \alpha\mathbf{r}) < \phi(\mathbf{x})$ bez indexování iterací.

$$\phi(\mathbf{x} + \alpha\mathbf{r}) = \frac{1}{2}(\mathbf{x} + \alpha\mathbf{r})^T\mathbf{A}(\mathbf{x} + \alpha\mathbf{r}) - (\mathbf{x} + \alpha\mathbf{r})^T\mathbf{b} \tag{3.20}$$

Po úpravách a dosazení za $\mathbf{x}$ z (3.18) dostaneme

$$\phi(\mathbf{x} + \alpha\mathbf{r}) = \phi(\mathbf{x}) - \alpha\mathbf{r}^T\mathbf{r} + \frac{1}{2}\alpha^2\mathbf{r}^T\mathbf{A}\mathbf{r}. \tag{3.21}$$

Funkcionál (3.21) minimalizujeme položením

$$\alpha = \frac{\mathbf{r}^T\mathbf{r}}{\mathbf{r}^T\mathbf{A}\mathbf{r}}, \quad \text{resp.} \quad \alpha_k = \frac{\mathbf{r}^{(k-1)^T}\mathbf{r}^{(k-1)}}{\mathbf{r}^{(k-1)^T}\mathbf{A}\mathbf{r}^{(k-1)}}. \tag{3.22}$$

Algoritmus metody největšího spádu můžeme tedy popsat takto:

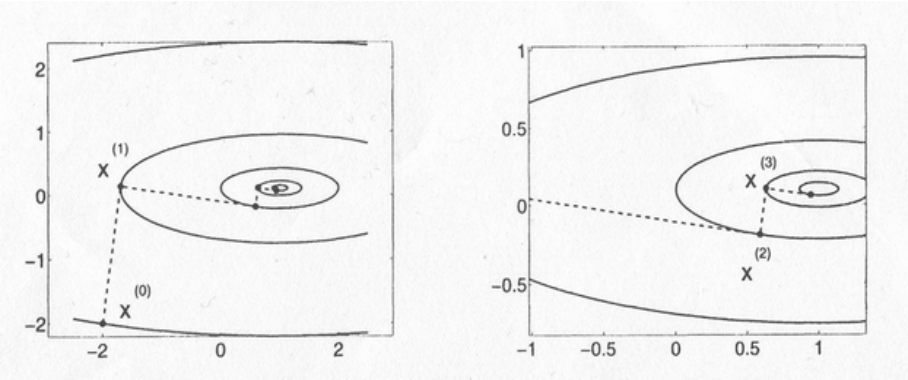
Aloritmus metody největšího spádu (v anglické literatuře označení The Method Of Steepest Descent)

$\mathbf{x}^{(0)}$ = počáteční aproximace
$\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(0)}$
$k = 0$
`while` $\mathbf{r}^{(k)} \neq 0$
$k = k + 1$
$\alpha_k = \mathbf{r}^{(k-1)^T}\mathbf{r}^{(k-1)} / \mathbf{r}^{(k-1)^T}\mathbf{A}\mathbf{r}^{(k-1)}$
$\mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} + \alpha_k \mathbf{r}^{(k-1)}$
$\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}$
`end`

Metoda největšího spádu s sebou přináší riziko v podobě tzv. „zig–zag“ efektu. Pokud bude číslo podmíněnosti $\kappa(A) = \frac{\lambda_{max}}{\lambda_{min}} = \frac{\lambda_1}{\lambda_n}$ vysoké, bude metoda největšího spádu konvergovat velmi pomalu. Jednoduchou geometrickou interpretaci ukážeme pro případ $n = 2$. Matici $\mathbf{A}$ odpovídají vlastní čísla λ_1 a λ_2, vektor $\mathbf{x} = (x_1, x_2)^T$. Křivky odpovídající $\phi(x_1, x_2) = c$, kde $c \in \mathbb{R}^+$, tvoří soustředné elipsy, velikost hlavní a vedlejší poloosy elips je nepřímo úměrná velikosti vlastních čísel λ_1, λ_2. Pokud bude $\lambda_1 = \lambda_2$, elipsy přejdou v kružnice a ve směru gradientu získáme v první iteraci hledané minimum. Pokud naopak bude $\lambda_1 >> \lambda_2$, budou elipsy zploštělé a metoda bude konvergovat velmi pomalu (směry gradientů budou tvořit ostře lomenou čáru, tj. „zig–zag“ efekt) – viz obrázek 3.1.

Metoda sdružených gradientů

Abychom se vyhnuli problémům spojeným s „zig–zag“ efektem, budeme hledat minimum $\phi(\mathbf{x})$ v jiném směru než ve směru residuí $\{\mathbf{r}^{(0)}, \mathbf{r}^{(1)}, \ldots\}$. Hledané směry označíme $\{\mathbf{p}^{(1)}, \mathbf{p}^{(2)}, \ldots\}$. Potom posloupnost hledaných $\mathbf{x}$ je



Obrázek 3.1: Princip „zig–zag“ efektu – převzato z [5]

dána vztahem

$$\mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} + \alpha_k \mathbf{p}^{(k)}. \tag{3.23}$$

Stejným způsobem, jakým jsme odvodili vztah (3.22) u metody největšího spádu, dostaneme

$$\alpha_k = \frac{\mathbf{p}^{(k)^T} \mathbf{r}^{(k-1)}}{\mathbf{p}^{(k)^T} \mathbf{A} \mathbf{p}^{(k)}}. \tag{3.24}$$

Otázkou je, jakým způsobem hledat směry $\mathbf{p}$. Ukazuje se vhodné volit hledané směry tak, aby byly navzájem sdružené vzhledem k matici $\mathbf{A}$. Hledaný směr bude ve tvaru

$$\mathbf{p}^{(k)} = \mathbf{r}^{(k-1)} + \beta_k \mathbf{p}^{(k-1)}, \tag{3.25}$$

kde $\beta_k \in \mathbb{R}$ musí splňovat

$$\mathbf{p}^{(k-1)^T} \mathbf{A} \mathbf{p}^{(k)} = 0. \tag{3.26}$$

Toto splňuje

$$\beta_k = -\frac{\mathbf{p}^{(k-1)^T} \mathbf{A} \mathbf{r}^{(k-1)}}{\mathbf{p}^{(k-1)^T} \mathbf{A} \mathbf{p}^{(k-1)}}. \tag{3.27}$$

Dostáváme první verzi algoritmu metody sdružených gradientů, kterou můžeme popsat následujícím způsobem:

Algoritmus metody sdružených gradientů – verze 1

$\mathbf{x}^{(0)}$ = počáteční aproximace
$k = 0$
$\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(0)}$
`while` $\mathbf{r}^{(k)} \neq 0$
    $k = k + 1$
    `if` $k = 1$
        $\mathbf{p}^{(1)} = \mathbf{r}^{(0)}$
    `else`
        $\beta_k = -(\mathbf{p}^{(k-1)^T}\mathbf{A}\mathbf{r}^{(k-1)})/(\mathbf{p}^{(k-1)^T}\mathbf{A}\mathbf{p}^{(k-1)})$
        $\mathbf{p}^{(k)} = \mathbf{r}^{(k-1)} + \beta_k\mathbf{p}^{(k-1)}$
    `end`
    $\alpha_k = (\mathbf{p}^{(k)^T}\mathbf{r}^{(k-1)})/(\mathbf{p}^{(k)^T}\mathbf{A}\mathbf{p}^{(k)})$
    $\mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} + \alpha_k\mathbf{p}^{(k)}$
    $\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}$
`end`

Pro zvýšení efektivity provedeme ve výše uvedeném algoritmu několik úprav. Pokud nahradíme vzorec pro výpočet residua rekurzivním vzorcem

$$\mathbf{r}^{(k)} = \mathbf{r}^{(k-1)} - \alpha_k\mathbf{A}\mathbf{p}^{(k)}$$

a ve vztahu pro výpočet β_k nahradíme

$$\mathbf{r}^{(k-1)^T}\mathbf{r}^{(k-1)} = -\alpha_{k-1}\mathbf{r}^{(k-1)^T}\mathbf{A}\mathbf{p}^{(k-1)}$$

$$\mathbf{r}^{(k-2)^T}\mathbf{r}^{(k-2)} = \alpha_{k-1}\mathbf{p}^{(k-1)^T}\mathbf{A}\mathbf{p}^{(k-1)},$$

dostaneme efektivní verzi algoritmu sdružených gradientů, která je podrobně popsána v [4]:

Algoritmus metody sdružených gradientů – verze 2 (v anglické literatuře The Conjugate Gradient Method)

$\mathbf{x}^{(0)}$ = počáteční aproximace ($\mathbf{A}\mathbf{x}^{(0)} \approx \mathbf{b}$)
$k = 0$
$\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(0)}$
`while` $\mathbf{r}^{(k)} \neq 0$
    $k = k + 1$
    `if` $k = 1$
        $\mathbf{p}^{(1)} = \mathbf{r}^{(0)}$
    `else`
        $\beta_k = (\mathbf{r}^{(k-1)^T}\mathbf{r}^{(k-1)})/(\mathbf{r}^{(k-2)^T}\mathbf{r}^{(k-2)})$
        $\mathbf{p}^{(k)} = \mathbf{r}^{(k-1)} + \beta_k\mathbf{p}^{(k-1)}$
    `end`
    $\alpha_k = (\mathbf{r}^{(k-1)^T}\mathbf{r}^{(k-1)})/(\mathbf{p}^{(k)^T}\mathbf{A}\mathbf{p}^{(k)})$
    $\mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} + \alpha_k\mathbf{p}^{(k)}$
    $\mathbf{r}^{(k)} = \mathbf{r}^{(k-1)} - \alpha_k\mathbf{A}\mathbf{p}^{(k)}$
`end`

Zbývá určit, jakým způsobem výpočet ukončíme. Kdybych se pohybovali v přesné aritmetice, konvergence metody sdružených gradientů by byla zajištěna v nejvýše n krocích a výpočet bychom zastavili v momentě, kdy $\mathbf{r}^{(k)} = \mathbf{0}$. Vlivem zaokrouhlovacích chyb však tato podmínka ztrácí na relevantnosti. Jedna z možností, jak ji nahradit, je zkoumat v každé iteraci normu residua a výpočet provádět do té doby, dokud platí

$$\|\mathbf{r}^{(k)}\|_2 = \sqrt{\mathbf{r}^{(k)^T}\mathbf{r}^{(k)}} > tol, \tag{3.28}$$

kde $\|\mathbf{r}^{(k)}\|_2$ je Eukleidovská norma vektoru $\mathbf{r}^{(k)}$ a tol je požadovaná přesnost řešení. Stejná situace nastává i pro případ metody největšího spádu, zastavovací podmínku pro ukončení výpočtu můžeme volit stejnou jako je rovnice (3.28).

Lze dokázat, že metoda sdružených gradientů konverguje velmi rychle, pokud je matice soustavy dobře podmíněná. Stejný závěr lze učinit i v případě, že matice soustavy je v určitém smyslu "blízká" jednotkové matici ("blízká"

ve smyslu normy nebo ve smyslu malých změn v prvcích oproti jednotkové matici[4]). Velké množství matic (například matice normálních rovnic) tyto vlastnosti nemá. V následující kapitole si ukážeme, jakým způsobem lze někdy řešení soustavy $\mathbf{Ax} = \mathbf{b}$ převést na řešení jiné soustavy $\tilde{\mathbf{A}}\tilde{\mathbf{x}} = \tilde{\mathbf{b}}$, kde matice $\tilde{\mathbf{A}}$ je blízká jednotkové matici.

[4]z anglického lower rank perturbation of the identity

Kapitola 4

Teoretická část – modifikovné metody

4.1 Předpodmínění

V této části si vysvětlíme, co je to tzv. předpodmínění, ukážeme si různé způsoby a druhy předpodmínění. Prakticky je aplikujeme na metodu největšího spádu a metodu sdružených gradientů, naším cílem bude získání efektivnějšího způsobu výpočtu soustavy normálních rovnic.

Chceme řešit soustavu (3.1), která má tvar $\mathbf{Ax} = \mathbf{b}$, kde matice $\mathbf{A}$ je symetrická a pozitivně definitní. Místo této soustavy budeme řešit soustavu

$$\tilde{\mathbf{A}}\tilde{\mathbf{x}} = \tilde{\mathbf{b}}, \tag{4.1}$$

kde $\tilde{\mathbf{A}} = \mathbf{C}^{-1}\mathbf{A}\mathbf{C}^{-1}$, $\tilde{\mathbf{x}} = \mathbf{Cx}$, $\tilde{\mathbf{b}} = \mathbf{C}^{-1}\mathbf{b}$, matice $\mathbf{C}$ je symetrická a pozitivně definitní. Matici $\mathbf{C}$ musíme volit tak, aby $\tilde{\mathbf{A}}$ byla dobře podmíněná nebo aby její vlastní čísla měla malý rozptyl (vlastnost odpovídající matici v jistém smyslu "blízké" jednotkové matici), čímž zajistíme rychlou konvergenci metody sdružených gradientů aplikované na soustavu (4.1). V podstatě se tedy snažíme nahradit řešení špatně podmíněné soustavy řešením jiné soustavy

vykazující lepší numerickou stabilitu. Soustavu (4.1) lze přepsat do tvaru

$$\mathbf{C}^{-1}\mathbf{A}\mathbf{C}^{-1}\mathbf{C}\mathbf{x} = \mathbf{C}^{-1}\mathbf{b}, \tag{4.2}$$

po úpravě

$$\mathbf{C}^{-1}\mathbf{A}\mathbf{x} = \mathbf{C}^{-1}\mathbf{b}. \tag{4.3}$$

Tuto rovnici nazýváme rovnicí předpodmínění, konkrétně předpodmínění zleva. Podobně bychom mohli uvažovat předpodmínění zprava nebo oboustranné[1] předpodmínění

$$\mathbf{A}\mathbf{C}^{-1}\mathbf{y} = \mathbf{b}, \quad \mathbf{y} = \mathbf{C}\mathbf{x} \tag{4.4}$$

a

$$\mathbf{C}_L^{-1}\mathbf{A}\mathbf{C}_P^{-1}\mathbf{y} = \mathbf{C}_L^{-1}\mathbf{b}, \quad \mathbf{y} = \mathbf{C}_P\mathbf{x}, \tag{4.5}$$

matice $\mathbf{C}_L$ a $\mathbf{C}_P$ jsou matice odpovídající předpodmínění zleva a zprava. V závislosti na konkrétním způsobu předpodmínění je maticí předpodmínění nazývána přímo matice $\mathbf{C}$ nebo jiná matice. Volba vhodného způsobu předpodmínění není lehká ani jednoznačná otázka, při špatně zvoleném předpodmínění se může stát, že dojde ke zpomalení konvergence výpočtu, tedy k pravému opaku toho, co je naším cílem. Předpodmíněním se zabývá například literatura [4], [5] a [6].

Druhy předpodmínění

Ukážeme několik variant předpodmínění, zaměříme se na některé konkrétní způsoby, vysvětlíme jejich princip a nakonec se pokusíme o jejich aplikaci na metodu největšího spádu a metodu sdružených gradientů:

1. Diagonální předpodmínění
2. Neúplná Choleského faktorizace
3. Polynomiální předpodmínění

[1] z anglického centered preconditioning

4.1.1 Diagonální předpodmínění

V případě, že matice $\mathbf{A}$ je symetrická a pozitivně definitní, lze podle [5] volit matici $\mathbf{C}$ jako diagonální matici, která má na diagonále prvky diagonály matice $\mathbf{A}$. V tomto případě se matice $\mathbf{C}$ nazývá maticí předpodmínění. Je možné použít diagonální předpodmínění formou předpodmínění zleva, zprava i oboustranné. V podstatě se tedy vlastní algoritmus metody nemění, pouze vstupní data jsou jiná.

4.1.2 Neúplná Choleského faktorizace

Řekli jsme, že místo soustavy (3.1) budeme řešit soustavu (4.1). Ukážeme si předpodmínění neúplnou Choleského faktorizací pro metodu největšího spádu a pro metodu sdružených gradientů. Podle literatury [4] a [5] se tento způsob předpodmínění jeví jako jeden z nejčastěji používaných. V knize [4] je tento způsob předpodmínění odvozen pro metodu sdružených gradientů, obdobným způsobem jej přímo v této práci odvodíme i pro metodu největšího spádu. Příslušnou metodu vždy aplikujeme na soustavu (4.1). Vyjdeme z algoritmů odvozených v předchozích kapitolách, místo vektorů a matic $\mathbf{A}$, $\mathbf{b}$, $\mathbf{x}^{(k)}$, $\mathbf{r}^{(k)}$ a $\mathbf{p}^{(k)}$ budeme uvažovat $\tilde{\mathbf{A}} = \mathbf{C}^{-1}\mathbf{A}\mathbf{C}^{-1}$, $\tilde{\mathbf{b}} = \mathbf{C}^{-1}\mathbf{b}$, $\tilde{\mathbf{x}}^{(k)} = \mathbf{C}\mathbf{x}^{(k)}$ a definujeme $\tilde{\mathbf{p}}^{(k)} = \mathbf{C}\mathbf{p}^{(k)}$. Potom je zřejmé, že $\tilde{\mathbf{r}}^{(k)} = \tilde{\mathbf{b}} - \tilde{\mathbf{A}}\tilde{\mathbf{x}}^{(k)} = \mathbf{C}^{-1}\mathbf{b} - \mathbf{C}^{-1}\mathbf{A}\mathbf{C}^{-1}\mathbf{C}\mathbf{x}^{(k)} = \mathbf{C}^{-1}\mathbf{r}^{(k)}$. Po dosazení do výše uvedených algoritmů metod největšího spádu a sdružených gradientů:

Algoritmus metody největšího spádu pro soustavu (4.1) – odvozený v této práci:

$\mathbf{x}^{(0)}$ = počáteční aproximace
$\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(0)}$
$k = 0$
`while` $\mathbf{C}^{-1}\mathbf{r}^{(k)} \neq 0$
$k = k + 1$
$\alpha_k = \left((\mathbf{C}^{-1}\mathbf{r}^{(k-1)})^T(\mathbf{C}^{-1}\mathbf{r}^{(k-1)})\right) / \left((\mathbf{C}^{-1}\mathbf{r}^{(k-1)})^T(\mathbf{C}^{-1}\mathbf{A}\mathbf{C}^{-1})(\mathbf{C}^{-1}\mathbf{r}^{(k-1)})\right)$

$\mathbf{Cx}^{(k)} = \mathbf{Cx}^{(k-1)} + \alpha_k \mathbf{C}^{-1}\mathbf{r}^{(k-1)}$

$\mathbf{C}^{-1}\mathbf{r}^{(k)} = \mathbf{C}^{-1}\mathbf{b} - (\mathbf{C}^{-1}\mathbf{A}\mathbf{C}^{-1})\mathbf{C}\mathbf{x}^{(k)}$

`end`

Algoritmus metody sdružených gradientů (verze 2) pro soustavu (4.1):

$\mathbf{x}^{(0)}$ = počáteční aproximace ($\mathbf{A}\mathbf{x}^{(0)} \approx \mathbf{b}$)

$k = 0$

$\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(0)}$

`while` $\mathbf{C}^{-1}\mathbf{r}^{(k)} \neq 0$

$k = k + 1$

`if` $k = 1$

$\mathbf{C}\mathbf{p}^{(1)} = \mathbf{C}^{-1}\mathbf{r}^{(0)}$

`else`

$\beta_k = \left((\mathbf{C}^{-1}\mathbf{r}^{(k-1)})^T(\mathbf{C}^{-1}\mathbf{r}^{(k-1)})\right) / \left((\mathbf{C}^{-1}\mathbf{r}^{(k-2)})^T(\mathbf{C}^{-1}\mathbf{r}^{(k-2)})\right)$

$\mathbf{C}\mathbf{p}^{(k)} = \mathbf{C}^{-1}\mathbf{r}^{(k-1)} + \beta_k \mathbf{C}\mathbf{p}^{(k-1)}$

`end`

$\alpha_k = \left((\mathbf{C}^{-1}\mathbf{r}^{(k-1)})^T(\mathbf{C}^{-1}\mathbf{r}^{(k-1)})\right) / \left((\mathbf{C}\mathbf{p}^{(k)})^T(\mathbf{C}^{-1}\mathbf{A}\mathbf{C}^{-1})(\mathbf{C}\mathbf{p}^{(k)})\right)$

$\mathbf{C}\mathbf{x}^{(k)} = \mathbf{C}\mathbf{x}^{(k-1)} + \alpha_k \mathbf{C}\mathbf{p}^{(k)}$

$\mathbf{C}^{-1}\mathbf{r}^{(k)} = \mathbf{C}^{-1}\mathbf{r}^{(k-1)} - \alpha_k(\mathbf{C}^{-1}\mathbf{A}\mathbf{C}^{-1})\mathbf{C}\mathbf{p}^{(k)}$

`end`

V tomto případě matici předpodmínění označíme $\mathbf{M}$, je odvozena pomocí výše uvedené matice $\mathbf{C}$ jako

$$\mathbf{M} = \mathbf{C}^2, \tag{4.6}$$

matice $\mathbf{M}$ je stejně jako matice $\mathbf{C}$ symetrická a pozitivně definitní. Dále zavedeme vektor $\mathbf{z}$, který získáme v každé iteraci tak, že budeme řešit

$$\mathbf{M}\mathbf{z}^{(k)} = \mathbf{r}^{(k)}, \tag{4.7}$$

po dosazení z (4.6)

$$\mathbf{C}\mathbf{C}\mathbf{z}^{(k)} = \mathbf{r}^{(k)}.$$

Odtud tedy

$$\mathbf{z}^{(k)} = \mathbf{C}^{-1}\mathbf{C}^{-1}\mathbf{r}^{(k)}. \tag{4.8}$$

Výše uvedené algoritmy upravíme, provedeme transpozici výrazů v závorkách, získané součiny upravíme[2] $(\mathbf{C}^{-1})^T\mathbf{C}^{-1} = \mathbf{C}^{-1}\mathbf{C}^{-1}$ a nahradíme ze vztahu (4.8), poslední dvě rovnice v algoritmech vynásobíme zleva maticí $\mathbf{C}^{-1}$, resp. $\mathbf{C}$. Získáme následující algoritmy, které řeší naši původní soustavu (3.1), tedy $\mathbf{Ax} = \mathbf{b}$. Všimněme si, že matici $\mathbf{C}$ není vůbec třeba explicitně počítat.

Algoritmus předpodmíněné metody největšího spádu (odvozený v této práci):

$\mathbf{x}^{(0)}$ = počáteční aproximace
$\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(0)}$
$k = 0$
`while` $\mathbf{r}^{(k)} \neq 0$
    Řešení soustavy $\mathbf{M}\mathbf{z}^{(k)} = \mathbf{r}^{(k)}$
    $k = k + 1$
    $\alpha_k = (\mathbf{r}^{(k-1)^T}\mathbf{z}^{(k-1)})/(\mathbf{z}^{(k-1)^T}\mathbf{A}\mathbf{z}^{(k-1)})$
    $\mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} + \alpha_k\mathbf{z}^{(k-1)}$
    $\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}$
`end`

Algoritmus předpodmíněné metody sdružených gradientů:

$\mathbf{x}^{(0)}$ = počáteční aproximace ($\mathbf{A}\mathbf{x}^{(0)} \approx \mathbf{b}$)
$k = 0$
$\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(0)}$
`while` $\mathbf{r}^{(k)} \neq 0$
    Řešení soustavy $\mathbf{M}\mathbf{z}^{(k)} = \mathbf{r}^{(k)}$
    $k = k + 1$

[2] Z vlastností inverze plyne pro libovolnou matici $\mathbf{F}$, že $(\mathbf{F}^{-1})^T = (\mathbf{F}^T)^{-1}$, v případě symetrické matice (tedy např. matice $\mathbf{C}$), kdy $\mathbf{F}^T = \mathbf{F}$, platí dokonce, že výraz pro transpozici inverze je roven $\mathbf{F}^{-1}$.

```
    if k = 1
        p^(1) = z^(0)
    else
        β_k = (r^(k-1)^T z^(k-1)) / (r^(k-2)^T z^(k-2))
        p^(k) = z^(k-1) + β_k p^(k-1)
    end
    α_k = (r^(k-1)^T z^(k-1)) / (p^(k)^T A p^(k))
    x^(k) = x^(k-1) + α_k p^(k)
    r^(k) = r^(k-1) − α_k A p^(k)
end
```

Zbývá vyřešit otázku, jakým způsobem určit matici $\mathbf{M}$. Přístup metody neúplné Choleského faktorizace spočívá v tom, že se snažíme najít dolní trojúhelníkovou matici $\mathbf{H}$, která je v jistém smyslu blízká matici $\mathbf{G}$, což je dolní trojúhelníková matice Choleského rozkladu matice $\mathbf{A}$. Matice předpodmínění je potom rovna

$$\mathbf{M} = \mathbf{H}\mathbf{H}^T. \tag{4.9}$$

Ukážeme si, proč je výhodné volit matici $\mathbf{M}$ zrovna tímto způsobem. Připomeňme si, že $\mathbf{M} = \mathbf{C}^2$, přičemž $\mathbf{C}$ je symetrická a pozitivně definitní matice. Dále uvažujme QR–rozklad matice $\mathbf{C}$ ve tvaru $\mathbf{C} = \mathbf{Q}\mathbf{H}^T$, kde $\mathbf{Q}$ je ortogonální matice a $\mathbf{H}^T$ je horní trojúhelníková matice ($\mathbf{H}$ je výše uvedená dolní trojúhelníková matice). S využitím vlastností ortogonálních matic a vlastností maticové inverze[3] můžeme psát

$$\begin{aligned}\tilde{\mathbf{A}} &= \mathbf{C}^{-1}\mathbf{A}\mathbf{C}^{-1} = (\mathbf{C}^T)^{-1}\mathbf{A}\mathbf{C}^{-1} = (\mathbf{H}\mathbf{Q}^T)^{-1}\mathbf{G}\mathbf{G}^T(\mathbf{Q}\mathbf{H}^T)^{-1} = \\ &= (\mathbf{Q}^T)^{-1}\mathbf{H}^{-1}\mathbf{G}\mathbf{G}^T(\mathbf{H}^T)^{-1}\mathbf{Q}^{-1} = \mathbf{Q}\mathbf{H}^{-1}\mathbf{G}\mathbf{G}^T(\mathbf{H}^T)^{-1}\mathbf{Q}^T \approx \mathbf{I}. \end{aligned} \tag{4.10}$$

Tedy čím lépe matice $\mathbf{H}$ aproximuje matici $\mathbf{G}$, tím více se matice $\tilde{\mathbf{A}}$ blíží jednotkové matici (součiny $\mathbf{H}^{-1}\mathbf{G}$ a $\mathbf{G}^T(\mathbf{H}^T)^{-1}$ se budou blížit jednotkové matici, v případě, že by byly rovny jednotkové matici, výraz (4.10) by byl

[3]Pro libovolné matice $\mathbf{K}$ a $\mathbf{L}$ platí $(\mathbf{K}\mathbf{L})^{-1} = \mathbf{L}^{-1}\mathbf{K}^{-1}$ a $(\mathbf{K}^{-1})^T = (\mathbf{K}^T)^{-1}$.

roven $\mathbf{Q}\mathbf{Q}^T = \mathbf{I}$ vzhledem k tomu, že $\mathbf{Q}$ je ortogonální). To, že matice $\tilde{\mathbf{A}}$ bude blízká jednotkové matici a tedy bude mít i malé číslo podmíněnosti, zajistí rychlou konvergenci metody sdružených gradientů, stejné vlastnosti budeme očekávat i od konvergence metody největšího spádu.

Jednoduchý, ale efektivní postup, jak určit matici $\mathbf{H}$ aproximující matici $\mathbf{G}$, je počítat zjednodušeným způsobem Choleského rozklad matice $\mathbf{A}$, a to tak, že pokud je některý prvek $a_{ij} = 0$, položíme odpovídající prvek h_{ij} automaticky také roven nule. To je výhodné zvláště v případě, že matice $\mathbf{A}$ je řídká, neboť matice $\mathbf{H}$ bude logicky také řídká. Počet operací nutných pro její výpočet se tak značně snižuje, což je pro efektivitu metody nezbytné. Matici předpodmínění $\mathbf{M}$ určíme snadno podle (4.9). Připomeňme, že matice $\mathbf{M}$ je symetrická a pozitivně definitní, a je zřejmé, že bude mít opět řídkou strukturu.

Nesmíme zapomenout na podstatný krok v tomto způsobu předpodmínění, a to řešení soustavy (4.7). Oproti základním algoritmům metody největšího spádu a metody sdružených gradientů totiž v každém výpočetním cyklu řešíme navíc další soustavu rovnic. Využijeme toho, že matice soustavy $\mathbf{M}$ se v iteracích nemění. Nabízí se možnost řešit ji použitím Choleského rozkladu, můžeme dokonce využít zjednodušený Choleského rozklad, kterým jsme matici $\mathbf{M}$ určili, tedy $\mathbf{M} = \mathbf{H}\mathbf{H}^T$. Do výpočtu vstupuje pouze matice $\mathbf{H}$ a vektor pravé strany rovnice. Protože již máme určený rozklad matice $\mathbf{M}$, je řešení této pomocné soustavy zredukováno na počítání přímé a zpětné substituce (viz princip Choleského rozkladu).

Spojením algoritmu předpodmíněné metody největšího spádu, resp. algoritmu předpodmíněné metody sdružených gradientů s výpočtem matice předpodmínění $\mathbf{M}$ pomocí neúplné Choleského faktorizace dostáváme velmi sofistikovaný způsob předpodmínění, v anglické liturtuře známý jako Incomplete Cholesky Factorization nebo Incomplete Cholesky Preconditioners.

4.1.3 Polynomiální předpodmínění

V předchozí kapitole jsme definovali systém předpodmínění ve tvaru $\mathbf{Mz} = \mathbf{r}$ (vztah (4.7)), matici $\mathbf{M}$ jsme nazvali maticí předpodmínění. Vektor $\mathbf{z}$, který vztah (4.7) řeší, aproximuje řešení rovnice $\mathbf{Az} = \mathbf{r}$, tedy matice $\mathbf{M}$ musí aproximovat matici $\mathbf{A}$. Vysvětlíme si myšlenku polynomiálního předpodmínění. Abychom získali hledaný vektor $\mathbf{z}$, provedeme nejprve jistý počet iterací nějaké stacionární iterační metody (metody, které se nazývají stacionární, jsme vysvětlili již dříve):

$$\mathbf{M}_1\mathbf{z}^{k+1} = \mathbf{N}_1\mathbf{z}^{(k)} + \mathbf{r}, \tag{4.11}$$

počáteční aproximaci volíme například $\mathbf{z}^{(0)} = \mathbf{0}$. Uvažujeme, že provedeme p kroků stacionární metody (4.11). Označíme $\mathbf{G} = \mathbf{M}_1^{-1}\mathbf{N}_1$. Po dosazení do vztahu (4.11) získáme v p-té iteraci

$$z^{(p)} = (\mathbf{I} + \mathbf{G} + \cdots + \mathbf{G}^{p-1})\mathbf{M}_1^{-1}\mathbf{r}, \tag{4.12}$$

v tomto případě však horní indexy matice $\mathbf{G}$ (indexy bez závorek) vyjadřují mocniny, ne iterace! Pokud označíme $\mathbf{M}^{-1} = (\mathbf{I}+\mathbf{G}+\cdots+\mathbf{G}^{p-1})\mathbf{M}_1^{-1}$, vztah (4.12) přejde do již známého tvaru $\mathbf{z} = \mathbf{M}^{-1}\mathbf{r}$, resp. $\mathbf{Mz} = \mathbf{r}$, matice $\mathbf{M}$ je matice předpodmínění. Volba matic $\mathbf{M}_1$, $\mathbf{N}_1$ a počtu iterací p je limitována tím, že požadujeme, aby matice $\mathbf{M}$ byla symetrická a pozitivně definitní. Protože $\mathbf{M}$ je polynomiální matice (polynomiální v $\mathbf{G}$), mluvíme o polynomiálním předpodmínění.

V knize [4] je na základě polynomiálního předpodmínění odvozena metoda, která spojuje klasické iterační metody a předpodmíněnou metodu sdružených gradientů. Uvažujeme iterační metodu ve tvaru

$$\mathbf{x}^{(k)} = \mathbf{x}^{(k-2)} + \omega_k(\gamma_k\mathbf{z}^{(k-1)} + \mathbf{x}^{(k-1)} - \mathbf{x}^{(k-2)}), \tag{4.13}$$

kde vektor $\mathbf{z}^{(k-1)}$ získáme jako řešení soustavy (4.7) ve tvaru $\mathbf{Mz}^{(k-1)} = \mathbf{r}^{(k-1)}$. Čísla ω_k a γ_k představující urychlovací parametry, které by měly zrychlit konvergenci metody. Spojíme-li vztah (4.13) s algoritmem předpodmíněné

metody sdružených gradientů, dostaneme následující algoritmus:

Algoritmus předpodmíněné metody sdružených gradientů s použitím urychlovacích parametrů:

$\mathbf{x}^{(-1)} = \mathbf{0}$, $\mathbf{x}^{(0)}$ = počáteční aproximace ($\mathbf{A}\mathbf{x}^{(0)} \approx \mathbf{b}$)
$k = 0$
$\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(0)}$
`while` $\mathbf{r}^{(k)} \neq 0$
    $k = k + 1$
    Řešení soustavy $\mathbf{M}\mathbf{z}^{(k-1)} = \mathbf{r}^{(k-1)}$
    $\gamma_{k-1} = (\mathbf{z}^{(k-1)^T}\mathbf{M}\mathbf{z}^{(k-1)})/(\mathbf{z}^{(k-1)^T}\mathbf{A}\mathbf{z}^{(k-1)})$
    `if` $k = 1$
        $\omega_1 = 1$
    `else`
    $\omega_k = \left(1 - (\gamma_{k-1}\mathbf{z}^{(k-1)^T}\mathbf{M}\mathbf{z}^{(k-1)})/(\gamma_{k-2}\mathbf{z}^{(k-2)^T}\mathbf{M}\mathbf{z}^{(k-2)}\omega_{k-1})\right)^{-1}$
    `end`
    $\mathbf{x}^{(k)} = \mathbf{x}^{(k-2)} + \omega_k(\gamma_{k-1}\mathbf{z}^{(k-1)} + \mathbf{x}^{(k-1)} - \mathbf{x}^{(k-2)})$
    $\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}$
`end`

Volba urychlovacích parametrů může mít významný dopad na výslednou rychlost konvergence metody. Kniha [4] neuvádí odvození výpočtu urychlovacích parametrů.

4.2 Jiné způsoby modifikace

V této části si uvedeme metodu, která je opět modifikací známé metody sdružených gradientů, nelze ji však zařadit mezi metody předpodmínění.

4.2.1 Metoda sdružených residuí

Tato metoda spojuje dva velmi zajímavé přístupy do jedné metody, výsledkem je pak elegantní modifikace metody sdružených gradientů.

Symetrické a pozitivně definitní matice svými vlastnostmi dovolují řadu operací, které by pro obecnou matici nebyly možné. Jednou z nich je odmocninový rozklad[4], který je sice definován pro symetrickou a pozitivně semidefinitní matici, ale je zřejmé, že bude platit i pro matici symetrickou a pozitivně definitní (všechny pozitivně definitní matice jsou zároveň pozitivně semidefinitní). Uvažujme tedy symetrickou a pozitivně semidefinitní matici $\mathbf{B} \in \mathbb{R}^{n\times n}$ a její příslušný Choleského rozklad $\mathbf{B} = \mathbf{G}\mathbf{G}^T$. Dále uvažujeme singulární rozklad matice $\mathbf{G}$ ve tvaru $\mathbf{G} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ a položíme $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{U}^T$. Potom matice $\mathbf{X}$ je symetrická a pozitivně semidefinitní a platí

$$\begin{aligned} \mathbf{B} = \mathbf{G}\mathbf{G}^T = (\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T)(\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T)^T = \\ = \mathbf{U}\mathbf{\Sigma}^2\mathbf{U}^T = (\mathbf{U}\mathbf{\Sigma}\mathbf{U}^T)(\mathbf{U}\mathbf{\Sigma}\mathbf{U}^T) = \mathbf{X}^2, \end{aligned} \tag{4.14}$$

matici X nazýváme odmocninovým rozkladem matice $\mathbf{B}$. Pokud tedy budeme uvažovat soustavu normálních rovnic ve tvaru $\mathbf{A}\mathbf{x} = \mathbf{b}$ (soustava (3.1)), příslušný odmocninový rozklad $\mathbf{X}$ bude roven $\mathbf{X} = \mathbf{A}^{1/2}$, matice $\mathbf{X}$ je symetrická a pozitivně definitní. Soustava (3.1) přejde do tvaru $\mathbf{A}^{1/2}\mathbf{A}^{1/2}\mathbf{x} = \mathbf{b}$, po úpravě

$$\mathbf{A}^{1/2}\mathbf{x} = \mathbf{A}^{-1/2}\mathbf{b}. \tag{4.15}$$

Řešení této soustavy a soustavy (3.1) je tedy ekvivalentní.

V dalším kroku řešíme soustavu (4.15) upravenou metodou sdružených gradientů, která je určena pro soustavy s nesymetrickou maticí soustavy. Pokud bychom uvažovali soustavu (3.1) s nesymetrickou maticí $\mathbf{A}$, vynásobením zleva maticí $\mathbf{A}^T$ získáme ekvivalentní soustavu

$$\mathbf{A}^T\mathbf{A}\mathbf{x} = \mathbf{A}^T\mathbf{b}. \tag{4.16}$$

[4] z anglického matrix square root

Tuto soustavu lze řešit klasickou metodou sdružených gradientů, s tím, že místo matice $\mathbf{A}$ uvažujeme matici $\mathbf{A}^T\mathbf{A}$ a residuum soustavy (4.16) je $\mathbf{A}^T$-násobek skutečného residua $\mathbf{b} - \mathbf{A}\mathbf{x}$.

Uvedené dvě myšlenky tedy spojíme. Použijeme metodu sdružených gradientů odvozenou pro soustavu (4.16), ale jako vstupní soustavu budeme uvažovat odpovídající upravenou soustavu (4.15)

$$\left(\mathbf{A}^{1/2}\right)^T \mathbf{A}^{1/2}\mathbf{x} = \left(\mathbf{A}^{1/2}\right)^T \mathbf{A}^{-1/2}\mathbf{b}. \tag{4.17}$$

Po dosazení a několika úpravách dostaneme následující algoritmus:

Algoritmus metody sdružených residuí (v anglické literatuře The Conjugate Residual Method):

$\mathbf{x}^{(0)}$ = počáteční aproximace ($\mathbf{A}\mathbf{x}^{(0)} \approx \mathbf{b}$)
$k = 0$
$\mathbf{r}^{(0)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(0)}$
`while` $\mathbf{r}^{(k)} \neq 0$
    $k = k + 1$
    `if` $k = 1$
        $\mathbf{p}^{(1)} = \mathbf{r}^{(0)}$
    `else`
        $\beta_k = \left(\mathbf{r}^{(k-1)^T}\mathbf{A}\mathbf{r}^{(k-1)}\right) / \left(\mathbf{r}^{(k-2)^T}\mathbf{A}\mathbf{r}^{(k-2)}\right)$
        $\mathbf{A}\mathbf{p}^{(k)} = \mathbf{A}\mathbf{r}^{(k-1)} + \beta_k\mathbf{A}\mathbf{p}^{(k-1)}$
    `end`
    $\alpha_k = \left(\mathbf{r}^{(k-1)^T}\mathbf{A}\mathbf{r}^{(k-1)}\right) / \left(\left(\mathbf{A}\mathbf{p}^{(k)}\right)^T\mathbf{A}\mathbf{p}^{(k)}\right)$
    $\mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} + \alpha_k\mathbf{p}^{(k)}$
    $\mathbf{r}^{(k)} = \mathbf{r}^{(k-1)} - \alpha_k\mathbf{A}\mathbf{p}^{(k)}$
`end`

Kapitola 5

Praktická část

Základem praktické části této práce je vlastní implementace vybraných metod. Programové řešení je realizováno v programu Matlab®. Příslušné zdrojové kódy jsou k dispozici na přiloženém CD, zdrojové kódy vybraných metod jsou uvedeny v příloze. Všechny metody, se kterými budeme dále pracovat, jsou dílem vlastní implementace. V nezbytných případech budeme pracovat s funkcemi již implementovanými v programu Matlab®, což bude v textu vždy označeno.

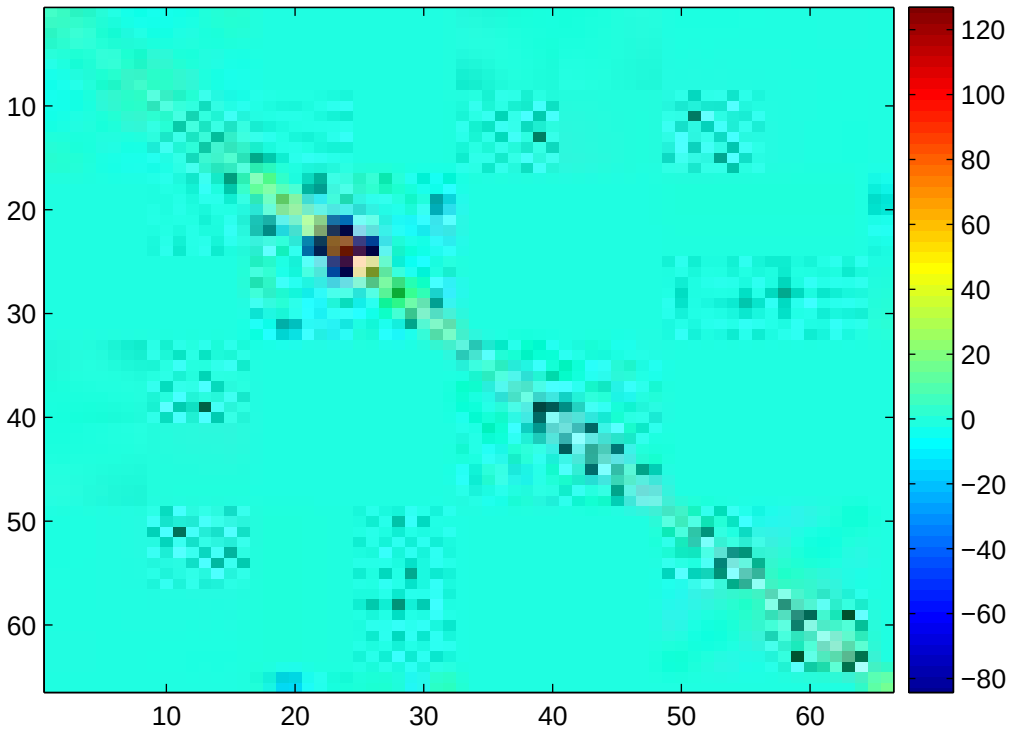
5.1 Testovací data

K praktickému zhodnocení implementovaných metod je potřeba provést testování algoritmů na reálných datech, která nejlépe simulují spojení s praxí. V našem případě budeme vyvinuté metody testovat na dvou soustavách normálních rovnic. Obě soustavy jsou soustavy normálních rovnic, které dostaneme při vyrovnání vázané plošné sítě pomocí zprostředkujících pozorování. Data jdoucí do vyrovnání jsou v obou případech data naměřená na geodetickém kursu v Nečtinech, který je součástí výuky oboru Geomatika na Západočeské univerzitě v Plzni. V obou případech se jedná o plošnou síť, která vznikla za účelem zhuštění základního polohového bodového pole v okolí Nečtin. Většinou bylo prováděno vedením polygonových pořadů a ob-

servacemi na známých i neznámých bodech.

Soustava 1

Testovací soustava je sestavena na základě měření z roku 2002. S těmito daty jsme pracovali v rámci semestrální práce [9] z předmětu Geodetické metody vyrovnání (KMA/GMV).



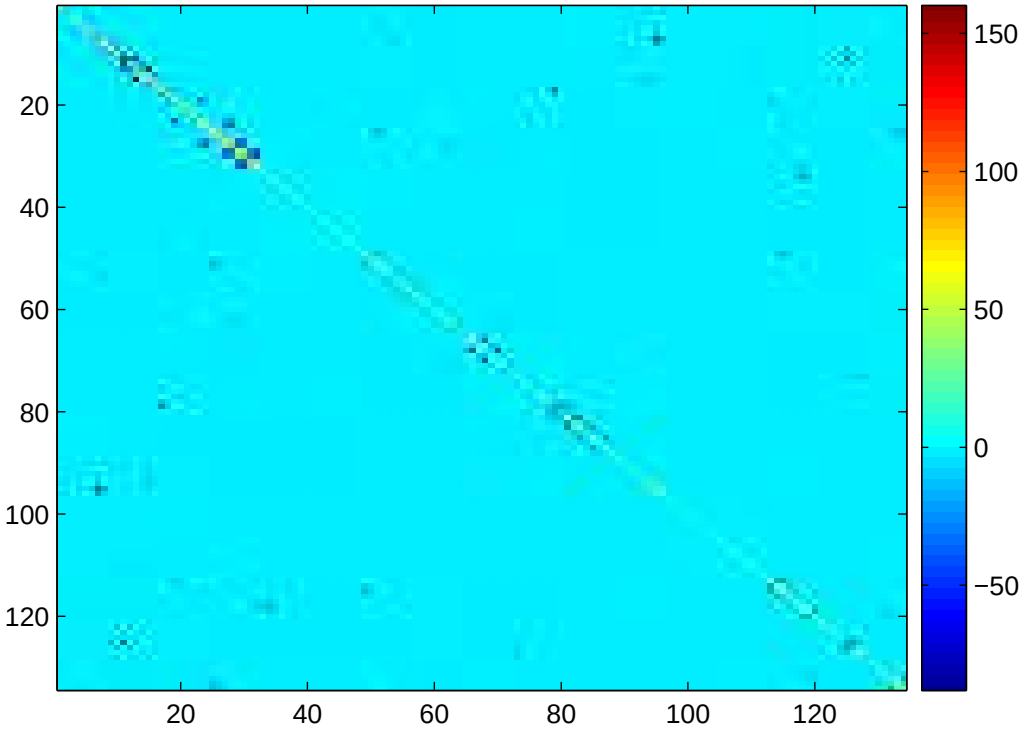
Obrázek 5.1: Rozložení prvků v matici normálních rovnic **N** první testovací soustavy

Matice normálních rovnic má rozměr 66×66. Rozložení a hodnoty prvků matice jsou vidět na obrázku. Hodnoty prvků jsou znázorněny barvou podle

barevné stupnice. Číslo podmíněnosti[1] matice je 1319, což ukazuje na špatné numerické vlastnosti prováděných výpočtů.

Soustava 2

Soustava vznikla na základě spojení dat naměřených v letech 2000 až 2003. Podrobnosti jsou uvedeny v [15].



Obrázek 5.2: Rozložení prvků v matici normálních rovnic **N** druhé testovací soustavy

Matice normálních rovnic má rozměr 134 × 134. Rozložení a hodnoty prvků

[1] Číslo podmíněnosti je určené pomocí funkce programu Matlab® `cond`, více informací o této funkci lze najít v [14].

matice jsou vidět na obrázku. Číslo podmíněnosti matice je 2737, je tedy ještě větší než číslo podmíněnosti matice první soustavy.

5.2 Přehled implementovaných metod

Uvedeme si přehled všech metod, se kterými budeme pracovat. V následujícím seznamu je vždy za pomlčkou uvedeno jméno příslušné funkce, kterou jsme pro danou metodu naprogramovali. V případě, že se implementace skládá z více funkcí, je uvedeno pouze jméno hlavní funkce.

- Klasické metody – obecné
  - Singulární rozklad pro symetrické matice (symetrický Schurův rozklad) – `singValDecomp`
    Jedná se o řešení soustavy normálních rovnic pomocí singulárního rozkladu matice, je realizován pro symetrickou matici soustavy. Singulární rozklad matice slouží jako referenční metoda vzhledem k metodám gradientního typu, představuje zástupce metod, které jsou pro svojí stabilitu a robustnost běžně používané.
- Klasické metody – gradientní
  - Metoda největšího spádu – `steepestDescent`
  - Metoda sdružených gradientů – `conjugateGradients`
    Dvě gradientní metody v základním tvaru, slouží jako referenční metody vzhledem k modifikovaným metodám. Porovnáním výsledků dosažených pomocí základních a modifikovaných metod lze zhodnotit zlepšení průběhu výpočtu.
- Modifikované metody
  - Metoda největšího spádu s použitím diagonálního předpodmínění – varianta zleva, oboustranné a zprava – `diagonalSD_l`, `diagonalSD_c`, `diagonalSD_r`

- Metoda sdružených gradientů s použitím diagonálního předpodmínění – varianta zleva, oboustranné a zprava – `diagonalCG_l`, `diagonalCG_c`, `diagonalCG_r`
- Metoda největšího spádu s použitím předpodmínění neúplnou Choleského faktorizací – `SD_chol`
- Metoda sdružených gradientů s použitím předpodmínění neúplnou Choleského faktorizací – `CG_chol`
- Metoda sdružených gradientů s použitím předpodmínění pomocí urychlovacího parametru – `conjugateGradientsOmega`
- Metoda sdružených residuí (modifikace metody sdružených gradientů) – `conjugateResiduals`

5.3 Přesnost řešení

Vektor řešení, který je výsledkem výpočtu soustavy normálních rovnic, představuje vektor oprav přibližných hodnot souřadnic hledaných bodů, opravy jsou v jednotce metr. Tyto opravy by se měly pohybovat v řádu centimetrů až decimetrů. Souřadnice bodů se určují s přesností na centimetry, tedy řádově 10^{-2} m. Budeme požadovat přesnost řešení o řád vyšší – 10^{-3} m, protože třetí desetinné místo již bude zaokrouhlené (kdybychom řešení počítali s přesností 10^{-2} m, bylo by již druhé desetinné místo zaokrouhlené). Požadovaná přesnost řešení představuje zároveň ukončovací podmínku iteračního výpočtu. Nabízí se tedy postup, že budeme v každé iteraci počítat residuum soustavy a pracovat s normou residua soustavy, viz vztah (3.28), hodnotu tol nastavíme právě na požadovanou přesnost, např. $tol = 0,001$. Takto lze pracovat s přesností a ukončením výpočtu ve všech použitých metodách s výjimkou metod s diagonálním předpodmíněním. Ostatní metody totiž nemění vstupní data jdoucí do výpočtu, princip diagonálního předpodmínění však spočívá v tom, že použité metody zůstávají stejné, ale vstupní hodnoty (matice soustavy a vektor pravé strany) jsou změněny. Proto v každé iteraci při výpočtu residua

musíme pracovat s původní soustavou, nikoli se soustavou modifikovanou.

Hodnotu požadované přesnosti řešení lze podle potřeby měnit. Za účelem porovnávání jednotlivých metod zavedeme následující označení:

- Normální přesnost
  Hodnotu *tol* nastavíme $tol = 0,001$, slouží pro standardní výpočet.
- Zvýšená přesnost
  Hodnotu *tol* nastavíme $tol = 0,00001$, výsledky s touto přesností budou sloužit jako referenční hodnoty pro porovnání s ostatními hodnotami, budeme je považovat za „správné“.

5.4 Způsoby porovnání a zhodnocení výsledků

V předchozí části jsme uvedli všechny metody, se kterými budeme pracovat. Ve většině případů budeme mezi sebou porovnávat gradientní metody ve své standardní podobě s modifikovanými gradientními metodami. Metoda singulárního rozkladu bude sloužit jako referenční metoda vzhledem k metodám gradientního typu. Je potřeba zvolit vhodný způsob porovnání jednotlivých metod a dosažených výsledků.

5.4.1 Počet iterací

Jelikož pracujeme s iteračními metodami, nabízí se možnost porovnávat efektivitu jednotlivých metod pomocí počtu iterací potřebných k výpočtu řešení se stejnou přesností. Tento test budeme provádět pouze pro gradientní metody (základní i modifikované).

Způsob hodnocení pomocí pouhého počtu iterací však není zcela objektivní. Problém nastává například u metod předpodmíněných neúplnou Choleského faktorizací a u metody sdružených gradientů s urychlovacím parametrem, kde uvnitř vlastního výpočetního cyklu počítáme určité kroky navíc oproti

standardním gradientním metodám, konkrétně řešení soustavy $\mathbf{Mz}^{(k)} = \mathbf{r}^{(k)}$. V případě metod s diagonálním předpodmíněním je zase nutno započítat cenu za výpočet inverzní matice $\mathbf{C}^{-1}$.

Sledování počtu iterací slouží jako hrubá představa o výpočetní náročnosti jednotlivých metod. Lze takto poměrně výstižně porovnávat metody, jejichž jádro (hlavní výpočetní cyklus) je velmi podobné a není třeba započítat žádné další operace navíc – například porovnání počtu iterací potřebných k výpočtu řešení se stanovenou přesností pomocí metody sdružených gradientů a metody sdružených residuí.

5.4.2 Počet operací

Velmi precizní metodou porovnávání jenotlivých metod je sledování počtu vykonaných operací. Z numerického hlediska se omezíme pouze na počet násobení a dělení. Výsledný počet operací závisí na počtu provedených iterací a na rozměru řešené soustavy. V tabulce (5.1) je uvedena cena za některé elementární operace, které nás budou zajímat (uvažujeme matice $\mathbf{A}$ a $\mathbf{B}$ rozměru $n \times n$, vektory $\mathbf{u}$ a $\mathbf{v}$ rozměru $n \times 1$ a libovolné reálné číslo c).

U řídkých matic lze v některých případech počet operací značně snížit, řídkost matice lze vhodným způsobem využít a pracovat pouze s nenulovými prvky. Příkladem může být násobení dvou matic, z nichž jedna je diagonální (viz tabulka 5.1), kdy počet operací je roven n^2, v případě, že by obě matice byly plné, počet operací je roven n^3.

5.4.3 Porovnání residuí

Výpočet residua $\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{Ax}^{(k)}$ slouží ke sledování rychlosti konvergence výpočtu, velikost normy residua slouží k ukončení výpočtu. Residuum nám může sloužit i k porovnávání jednotlivých metod mezi sebou. Jednu z metod zvolíme jako referenční. Protože zkoumáme hlavně vlastnosti modifikovaných

Prováděná operace	Počet operací
c*$\mathbf{u}$	n
c*$\mathbf{A}$	n^2
$\mathbf{u}^T$*$\mathbf{v}$	n
$\mathbf{u}^T$*$\mathbf{A}$	n^2
$\mathbf{A}$*$\mathbf{u}$	n^2
$\mathbf{A}$*$\mathbf{B}$	n^3
$\mathbf{A}$*$\mathbf{u}$ ($\mathbf{A}$ diagonální)	n
$\mathbf{A}$*$\mathbf{B}$ ($\mathbf{A}$ diagonální, $\mathbf{B}$ plná)	n^2
$\mathbf{A}^{-1}$ ($\mathbf{A}$ plná)	n^3
$\mathbf{A}^{-1}$ ($\mathbf{A}$ diagonální)	n
$\mathbf{A}\mathbf{A}^T$ ($\mathbf{A}$ dolní trojúhelníková)	$\frac{n^3}{3}$

Tabulka 5.1: Počty vykonaných násobení a dělení v některých elementárních operacích

metod, zvolíme jako referenční metodu klasickou metodu největšího spádu nebo metodu sdružených gradientů. Tu budeme počítat ve zvýšené přesnosti. Porovnávané metody budeme počítat v normální přesnosti. Můžeme spočítat rozdíl vektorů řešení referenční a porovnávané metody. Vektor tohoto rozdílu označíme $\mathbf{q}$. Budeme sledovat následující hodnoty:

- Výpočet normy $\|\mathbf{q}\|_2 = \sqrt{\mathbf{q}^T\mathbf{q}}$.
- Průměrná hodnota prvku q_i.
- Prvek s maximální odchylkou od správného řešení $\max|q_i|$.

5.4.4 Doba výpočtu

Další možností, jak zhodnotit náročnost výpočtu, je měřit čas potřebný k provedení konkrétní operace. Zvolíme pro všechny metody stejnou požadovanou přesnost řešení a změříme dobu výpočtu pomocí jednotlivých metod.

Na základě získaných časových údajů lze jednotlivé metody porovnat. Tento druh testování je v jistém smyslu důslednější než apriorní odhad počtu operací. Má však i své nevýhody. Nelze jej použít obecně, dá se použít pouze ve spojení s výpočtem reálných dat. Vzhledem k tomu, že počítá čas celého průběhu výpočtu, vyžaduje zcela vyladěný kód implementovaný bez redundantních příkazů. Pro jeho realizaci lze využít funkce `tic` a `toc`, které jsou implementované v programu Matlab®. Měří čas, který je potřeba k vykonání určité sekvence příkazů. Funkce `tic` spouští měření času v místě, kde příkaz uvedeme. Funkce `toc` ukončí měření času, opět v místě, na kterém příkaz uvedeme, a vypíše hodnotu času v sekundách, případně ji uloží do proměnné. Více informací o použití obou příkazů lze nalézt v [14].

5.4.5 Test s Hilbertovou maticí

Na závěr provedeme výpočetní experiment. Tento test již není počítán na reálných datech. Není tedy pro tuto práci klíčový, protože nesimuluje chování implementovaných metod ve spojení s reálnými daty. Soustavy normálních rovnic jsou často špatně podmíněné. Čísla podmíněnosti testovacích soustav v této práci jsou 1319 a 2737, u rozsáhlejších soustav jsou pak čísla podmíněnosti i vyšší. Provedeme tedy experiment, ve kterém budeme pracovat se soustavou rovnic s vysokým číslem podmíněnosti.

Uvažujme matici $\mathbf{H}$ rozměru $n \times n$, jednotlivé její prvky jsou definovány

$$h_{ij} = \frac{1}{i+j-1} \quad \forall i, j = 1, 2, \ldots, n.$$

Tato matice je nazývána Hilbertova matice a je s oblibou uváděna jako příklad matice s vysokým číslem podmíněnosti (Hilbertova matice řádu 5 má číslo podmíněnosti $4,7661 \cdot 10^5$!, s řádem matice se zvyšuje i číslo podmíněnosti). Hilbertovu matici lze automaticky generovat v programu Matlab® pomocí funkce `hilb`.

Vytvoříme testovací soustavu lineárních algebraických rovnic s Hilbertovou maticí řádu 20. Vektor pravé strany budeme generovat tak, aby jeho prvky byly součty prvků odpovídajícího řádku matice soustavy. Potom je jasné, že vektor řešení bude vektor ve tvaru $\mathbf{x} = (1, 1, 1, \ldots, 1)^T$. Tato volba je výhodná, protože předem známe hodnotu správného řešení a můžeme snadno zhodnotit přesnost vypočítaných výsledků. Výpočet řešení provedeme všemi metodami.

Kapitola 6

Výsledky

6.1 Dílčí výsledky

6.1.1 Počet iterací

Metoda	**Počet iterací Soustava 1**	**Počet iterací Soustava 2**
největší spád	1989	1694
sdružené gradienty	56	78
nejv. spád – diagonální zleva	1725	975
nejv. spád – diagonální oboustr.	11243	19073
nejv. spád – diagonální zprava	1307	866
sdr. grad. – diagonální zleva	945	431
sdr. grad. – diagonální oboustr.	93	144
sdr. grad. – diagonální zprava	702	150
nejv. spád – Cholesky	76	63
sdr. grad. – Cholesky	14	14
sdr. grad. – urychlovací param.	14	14
sdružená residua	55	62

Tabulka 6.1: Počet potřebných iterací

Tabulka 6.1 ukazuje počty iterací potřebných pro získání řešení různými metodami, požadovaná přesnost řešení je $tol = 0,001$. Výpočet jsme provedli pro obě testovací soustavy. Z tabulky je patrné, že počet iterací vlastní metody sdružených gradientů je velmi nízký. Následné modifikace se potom ve svých výsledcích liší. V některých případech došlo ke zvýšení počtu iterací, v jiných ke snížení. Velmi zajímavé výsledky dostáváme pro metody modifikované neúplnou Choleského faktorizací a pro metodu sdružených gradientů s urychlovacím parametrem. Zde je snížení počtu iterací nejvýraznější. Počet iterací metody největšího spádu ve své základní podobě je poměrně vysoký, kromě oboustranného diagonálního předpodmínění dochází modifikacemi k jejich výraznému snížení.

Tento způsob hodnocení však podává pouze hrubou představu o celkové výpočetní náročnosti metod. Musíme brát v úvahu, že případné snížení počtu iterací některou z modifikací je vyváženo jistými kroky navíc. K získání komplexnějšího pohledu je potřeba určit, jak drahé je snížení počtu iterací, zda se vůbec vyplatí. Touto problematikou se budeme zabývat v následující části.

6.1.2 Počet operací

Uvažujeme soustavu s rozměrem matice $n \times n$, k nechť označuje počet iterací hlavního výpočetního cyklu, r je podíl nenulových prvků v matici normálních rovnic a počítáme jej jako podíl počtu nenulových prvků matice ku počtu všech prvků matice. Konstantu r využijeme při určování počtu operací metod s neúplným Choleského předpodmíněním a u metody sdružených gradientů s urychlovacím parametrem.

V tabulce 6.2 je uveden počet matematických operací násobení a dělení, který je potřebný k získání řešení se stanovenou přesností $tol = 0,001$. Výpočetní složitost všech metod kromě metod pracujících s neúplným Choleského rozkladem se pohybuje řádově v počtu n^2 operací, což jsme apriorně očekávali. U metod předpodmíněných neúplným Choleského rozkladem a u metody

s urychlovacím parametrem dosahuje výpočetní složitost řádu n^3, ale počet operací závisí na zaplnění matice soustavy. Podíl nenulových prvků matice soustavy r je pro obě soustavy uveden v hlavičce tabulky 6.3.

Metoda	$\mathbf{n^3}$	$\mathbf{n^2}$	**n**
největší spád	–	$2k+1$	$4k+1$
sdružené gradienty	–	$k+1$	$8k-2$
nejv.sp. – diag.l.	–	$3k+2$	$4k+3$
nejv.sp. – diag.ob.	–	$3k+3$	$5k+3$
nejv.sp. – diag.p.	–	$3k+2$	$5k+2$
sdr.gr. – diag.l.	–	$2k+2$	$8k$
sdr.gr. – diag.ob.	–	$2k+3$	$9k$
sdr.gr. – diag.p.	–	$2k+2$	$9k-1$
nejv.sp. – Chol.	$\frac{r}{4}+\frac{1}{3}$	$3k+\frac{r}{2}+1$	$5k+1$
sdr.gr. – Chol.	$\frac{r}{4}+\frac{1}{3}$	$2k+\frac{r}{2}+2$	$9k-3$
sdr.gr. – urychl.	$\frac{r}{4}+\frac{1}{3}$	$5k+\frac{r}{2}$	$10k-3$
sdr.residua	–	$3k$	$7k-1$

Tabulka 6.2: Počty operací – obecné hodnoty

Po dosazení hodnot k a n, získáme konkrétní počty iterací pro obě soustavy (viz tabulka 6.3). Podíváme se nejprve na modifikované metody největšího spádu. Počet operací je pro klasickou metodu největšího spádu velmi vysoký, je to způsobeno příliš velkým počtem iterací této metody. V modifikacích pak kromě případu oboustranného diagonálního předpodmínění došlo ve všech případech ke snížení původního počtu operací. Nejnižších hodnot dosahuje počet operací metody největšího spádu s neúplnou Choleského faktorizací.

Metoda sdružených gradientů ve své základní podobě dává velice dobré výsledky, počet potřebných operací je velmi nízký. Metoda sdružených gradientů s neúplnou Choleského faktorizací tento počet ještě snižuje. Ostatní me-

tody vyžadují větší počet operací, poměrně dobrých výsledků dosahuje metoda sdružených gradientů s urychlovacím parametrem a metoda sdružených residuí.

Metoda	**Soustava 1** $n = 66$ $r = 0,159$	**Soustava 2** $n = 134$ $r = 0,088$
největší spád	$17,86 \cdot 10^6$	$61,76 \cdot 10^6$
sdružené gradienty	$0,28 \cdot 10^6$	$1,50 \cdot 10^6$
nejv.sp. – diag.l.	$23,01 \cdot 10^6$	$53,08 \cdot 10^6$
nejv.sp. – diag.ob.	$150,65 \cdot 10^6$	$1040,26 \cdot 10^6$
nejv.sp. – diag.p.	$17,52 \cdot 10^6$	$47,27 \cdot 10^6$
sdr.gr. – diag.l.	$8,74 \cdot 10^6$	$15,98 \cdot 10^6$
sdr.gr. – diag.ob.	$0,88 \cdot 10^6$	$5,40 \cdot 10^6$
sdr.gr. – diag.p.	$6,54 \cdot 10^6$	$5,60 \cdot 10^6$
nejv.sp. – Chol.	$1,13 \cdot 10^6$	$4,31 \cdot 10^6$
sdr.gr. – Chol.	$0,25 \cdot 10^6$	$1,41 \cdot 10^6$
sdr.gr. – urychl.	$0,42 \cdot 10^6$	$2,13 \cdot 10^6$
sdr.residua	$0,74 \cdot 10^6$	$3,40 \cdot 10^6$

Tabulka 6.3: Počty operací – konkrétní hodnoty

6.1.3 Porovnání residuí

Tento způsob porovnání výsledků neslouží k vyvozování závěrů o metodách jako takových. Slouží spíše jako kontrolní srovnání přesností výsledků získaných jednotlivými metodami. Přesnost řešení je realizována stanovením ukončovací podmínky. Způsob ukončení výpočetního cyklu iteračních metod se může lišit. Pokud chceme porovnávat různé iterační metody mezi sebou, musí být výpočetní cyklus všech metod ukončen stejně. V každé iteraci se zkoumá, zda je splněna podmínka, kterou si předem stanovíme, cyklus se opakuje dokud je podmínka splněna. V této práci realizujeme ukončovací

podmínku pomocí normy residua. Residua mohou být počítána různými způsoby.

Metoda	**Residuum pro ukončení**
největší spád	$\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}$
sdružené gradienty	$\mathbf{r}^{(k)} = \mathbf{r}^{(k-1)} - \alpha_k \mathbf{A}\mathbf{p}^{(k)}$
nejv. spád – diagonální zleva	$\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}$ $\star$
nejv. spád – diagonální oboustr.	$\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}$ $\star$
nejv. spád – diagonální zprava	$\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}$ $\star$
sdr. grad. – diagonální zleva	$\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}$ $\star$
sdr. grad. – diagonální oboustr.	$\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}$ $\star$
sdr. grad. – diagonální zprava	$\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}$ $\star$
nejv. spád – Cholesky	$\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}$
sdr. grad. – Cholesky	$\mathbf{r}^{(k)} = \mathbf{r}^{(k-1)} - \alpha_k \mathbf{A}\mathbf{p}^{(k)}$
sdr. grad. – urychlovací param.	$\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}$
sdružená residua	$\mathbf{r}^{(k)} = \mathbf{r}^{(k-1)} - \alpha_k \mathbf{A}\mathbf{p}^{(k)}$

Tabulka 6.4: Způsoby výpočtu residuí

V tabulce 6.4 jsou uvedeny způsoby počítání residua, které jsou použity v jednotlivých metodách. Residuum počítané rekurzivně se však od standardně počítaného residua $\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}$ téměř neliší, shodují se přibližně do řádu 10^{-14}, což jsme zjistili zkušebním výpočtem s oběma residui zároveň a jejich porovnáváním v každé iteraci. Označení $\star$ u diagonálně předpodmíněných metod znamená, že uvedený způsob výpočtu residua byl přidán do výpočetního cyklu navíc. Tyto metody totiž pracují s jinými vstupními daty (modifikuje se samotná vstupní soustava), ale pro ukončení výpočtu je třeba pracovat s původní soustavou. Tento postup je zaplacen operacemi navíc (dopočítání vektoru řešení $\mathbf{x}$ původní soustavy v každém kroku, následný výpočet residua původní soustavy), ale pro objektivní hodnocení přesnosti metod je nezbytný.

Tabulky 6.5 a 6.6 ukazují, jak se liší vektory řešení získané jednotlivými metodami při nastavení stejné přesnosti řešení. Řešení počítáme v základní přesnosti $tol = 0,001$ a porovnáváme je vždy s řešením ve zvýšené přesnosti $tol = 0,00001$ získaného metodou největšího spádu, resp. metodou sdružených gradientů. Toto řešení ve zvýšené přesnosti považujeme za správné. Ačkoliv je všemi metodami spočítáno řešení se stejnou přesností, řešení určené modifikovanými metodami největšího spádu s diagonálním předpodmíněním se od skutečného řešení liší nejvíce. Poměrně vysokých hodnot dosahují i odchylky metody sdružených residuí.

	Soustava 1			Soustava 2		
Metoda	$\|\mathbf{q}\|_2$	$\bar{q}_i$	$\max\|q_i\|$	$\|\mathbf{q}\|_2$	$\bar{q}_i$	$\max\|q_i\|$
nejv.sp.-diag.l.	0,00228	0,00019	0,00077	0,00354	0,00017	0,00122
nejv.sp.-diag.ob.	0,00228	0,00020	0,00078	0,00359	0,00017	0.00120
nejv.sp.-diag.p.	0,00275	0,00024	0,00093	0,00457	0,00023	0.00155
nejv.sp.-Chol.	0,00076	0,00008	0,00018	0,00255	0,00015	0.00084

Tabulka 6.5: Residua řešení metody největšího spádu a modifikovaných metod

	Soustava 1			Soustava 2		
Metoda	$\|\mathbf{q}\|_2$	$\bar{q}_i$	$\max\|q_i\|$	$\|\mathbf{q}\|_2$	$\bar{q}_i$	$\max\|q_i\|$
sdr.gr.-diag.l.	0,00025	0,00003	0,00007	0,00032	0,00002	0,00007
sdr.gr.-diag.ob.	0,00036	0,00003	0,00010	0,00102	0,00007	0,00022
sdr.gr.-diag.p.	0,00020	0,00002	0,00006	0,00081	0,00005	0,00024
sdr.gr.-Chol.	0,00037	0,00004	0,00010	0,00068	0,00004	0,00019
sdr.gr.-urychl.	0,00037	0,00004	0,00010	0,00068	0,00004	0,00019
sdr. residua	0,00106	0,00010	0,00035	0,00506	0,00025	0,00177

Tabulka 6.6: Residua řešení metody sdružených gradientů a modifikovaných metod

6.1.4 Doba výpočtu

Čas, který je potřeba k výpočtu řešení pomocí jedné metody, není shodný při každém spuštění programu. Doba vykonávání stejného postupu mírně kolísá. Aby naměřená hodnota času co nejlépe vystihovala skutečnost, změříme spuštění každé metody stokrát, z naměřených časů spočítáme průměr. Pro porovnání celý výpočet provedeme na dvou počítačích různé výkonnosti (viz tabulka 6.7). U všech metod volíme stejnou požadovanou přesnost řešení $tol = 0,001$.

Parametry	Počítač 1	Počítač 2
CPU	400 MHz	1,79 GHz
operační paměť	192 MB RAM	512 MB RAM

Tabulka 6.7: Parametry počítačů použitých pro měření času

Metoda	**Čas – 1 [s]**	**Čas – 2 [s]**
singulární rozklad	15	2,3
největší spád	0,72	0,063
sdružené gradienty	0,045	0,0032
nejv. spád – diagonální zleva	1,1	0,089
nejv. spád – diagonální oboustr.	11	0,82
nejv. spád – diagonální zprava	1,3	0,10
sdr. grad. – diagonální zleva	0,95	0,079
sdr. grad. – diagonální oboustr.	0,16	0,019
sdr. grad. – diagonální zprava	0,78	0,065
nejv. spád – Cholesky	1,1	0,22
sdr. grad. – Cholesky	0,33	0,075
sdr. grad. – urychlovací param.	0,35	0,076
sdružená residua	0,048	0,0045

Tabulka 6.8: Doba výpočtu jednotlivých metod pro soustavu 1

Metoda	**Čas – 1 [s]**	**Čas – 2 [s]**
singulární rozklad	404	66
největší spád	3,5	0,17
sdružené gradienty	0,14	0,0088
nejv. spád – diagonální zleva	3,5	0,16
nejv. spád – diagonální oboustr.	68	3,3
nejv. spád – diagonální zprava	3,4	0,17
sdr. grad. – diagonální zleva	1,5	0,087
sdr. grad. – diagonální oboustr.	0,99	0,069
sdr. grad. – diagonální zprava	0,73	0,048
nejv. spád – Cholesky	0,65	0,10
sdr. grad. – Cholesky	0,21	0,038
sdr. grad. – urychlovací param.	0,26	0,044
sdružená residua	0,21	0,013

Tabulka 6.9: Doba výpočtu jednotlivých metod pro soustavu 2

V tabulkách 6.8 a 6.9 jsou uvedeny dosažené časy výpočtů. Nejkratší doba výpočtu odpovídá klasické metodě sdružených gradientů. Žádný z časů jejích modifikaci není kratší. Nejvíce se k času metody sdružených gradientů přibližuje metoda sdružených residuí, v případě druhé testovací soustavy dostáváme velmi dobré výsledky i pro metodu sdružených gradientů s Choleského faktorizací a pro metodu sdružených gradientů s urychlovacím parametrem. V případě první testovací soustavy jsou však časy dosažené těmito metodami poměrně vysoké.

Doba výpočtu klasické metody největšího spádu se oproti metodě sdružených gradientů samozřejmě liší, odpovídající modifikace jsou tedy také obecně pomalejší. Ke zrychlení výpočtu metody největšího spádu dochází pouze pro Choleského modifikaci, a to v případě druhé testovací soustavy. Metody s Choleského modifikací a metoda s urychlovacím parametrem jsou para-

doxně pomalejší pro první testovací soustavu, která je rozměrově menší.

Nevýhodou tohoto druhu porovnání je, že neznáme přesný princip, jak měření času probíhá. Na samotnou hodnotu času má vliv mnoho faktorů, z nichž ne všechny byly v této práci detailně řešeny. Měl by se zohlednit například způsob uložení prvků v maticích (uchovávání pouze nezbytného počtu prvků co nejúspornějším způsobem), náročnost uložení na paměť, atd. Při vlastní implementaci jsme se zaměřili hlavně na zamezení vzniku redundantních příkazů.

6.1.5 Test s Hilbertovou maticí

Číslo podmíněnosti Hilbertovy matice řádu 20 je rovno $1,9084 \cdot 10^{18}$. V tabulce 6.10 jsou uvedeny hodnoty vektorů řešení určených jednotlivými metodami, správné řešení je $\mathbf{x} = (1, 1, 1, \ldots, 1)^T$. V tabulce nejsou uvedeny výsledky určené metodou singulárního rozkladu, ta v průběhu výpočtu selhává, stejně jako všechny metody diagonálně předpodmíněné. Metody největšího spádu a sdružených gradientů s Choleského předpodmíněním a metoda sdružených gradientů s urychlovacím parametrem potřebují k získání řešení jednu iteraci, vektor řešení však zcela neodpovídá skutečné hodnotě. Velikost residua $\mathbf{r} = \mathbf{b} - \mathbf{Ax}$ však odpovídá požadované přesnosti řešení $tol = 0,0001$! V tabulce je pro srovnání uvedeno řešení získané pomocí funkce pro řešení soustav rovnic, která je již naprogramovaná v programu Matlab® (zápis funkce – zpětné lomítko „\“). Pracuje na principu Gaussovy eliminace (viz [14]). Rovněž tato metoda poskytuje špatné výsledky. Výpis řešení této metody je doprovázen hlášením, které upozorňuje na přiliš vysoké číslo podmíněnosti. Z výsledků je patrné, že nejsprávnější řešení dává metoda největšího spádu, metoda sdružených gradientů a metoda sdružených residuí, což ukazuje na jejich vysokou stabilitu.

nejv. spád	**sdruž. grad.**	**nejv.sp. Chol.**	**sdr.gr. Chol.**	**sdr.gr. ur.par.**	**sdruž. res.**	**Matlab® „\“**
0,99675	0,99420	1,0006	1,0006	1,0006	0,99426	1
1,0241	1,0389	0,93356	0,93356	0,93356	1,0387	1
0,98610	0,97792	2,7414	2,7414	2,7414	0,97789	0,99969
0,97584	0,96493	-13,695	-13,695	-13,695	0,96499	1,0039
0,98098	0,97405	-9,6772	-9,6772	-9,6772	0,97413	0,97936
0,99156	0,98942	963,15	963,15	963,15	0,98952	1,0126
1,0025	1,0042	-7225,3	-7225,3	-7225,3	1,0043	1,3920
1,0117	1,0158	28339	28339	28339	1,0159	-1,1443
1,0181	1,0236	-68834	-68834	-68834	1,0236	6,6138
1,0218	1,0276	108920	108920	108920	1,0276	-7,869
1,0227	1,0282	-112670	-112670	-112670	1,0282	11,931
1,0213	1,0260	73409	73409	73409	1,0260	-15,236
1,0179	1,0214	-26959	-26959	-26959	1,0214	26,194
1,0127	1,0149	3767,8	3767,8	3767,8	1,0149	-24,907
1,0061	1,0068	294,95	294,95	294,95	1,0068	16,506
0,99833	0,99755	18,086	18,086	18,086	0,99750	-10,456
0,98961	0,98731	309,78	309,78	309,78	0,98724	19,552
0,98013	0,97633	-582,93	-582,93	-582,93	0,97625	-18,490
0,97005	0,96480	367,28	367,28	367,28	0,96472	10,857
0,95952	0,95287	-80,328	-80,328	-80,328	0,95278	-0,93904

Tabulka 6.10: Vektory řešení soustavy s Hilbertovou maticí

6.2 Celkové porovnání

Provedli jsme porovnání a zhodnocení dosažených výsledků podle jednotlivých kritérií. Na základě dílčích závěrů se nyní pokusíme o komplexnější analýzu.

Podle porovnání pouhého počtu iterací bychom mohli soudit, že většina modifikovaných metod vede ke zrychlení výpočtu. Potřebný počet iterací se totiž snižuje, a to i u metody sdružených gradientů, která ve své klasické podobě sama o sobě poskytuje velmi dobré výsledky. Pouhý počet iterací však není objektivním hodnocením. Je pravda, že modifikacemi jsme dosáhli snížení počtu iterací, ale cena za jeden výpočetní cykus vzrostla. V každé iteraci provádíme výpočetní kroky navíc, které zvyšují cenu za jednu iteraci. Záleží na tom, zda je snížení počtu iterací tak výrazné, že převáží zvýšení výpočetní složitosti jedné iterace. V případě, že je snížení počtu iterací menší, nesmí být výpočetní cena za jeden cyklus příliš vysoká. K přesnější analýze výpočetní složitosti slouží sledování počtu prováděných operací, který podává komplexní obraz efektivity jednotlivých metod. V prvním kroku je teoreticky určen potřebný počet operací, následně je dosazeno do obecných vztahů, čímž dostáváme cenu za výpočet jedné soustavy jednou konrétní metodou.

Nejnižší počet operací je potřeba k výpočtu pomocí metody sdružených gradientů s neúplným Choleského předpodmíněním. Téměř shodný počet operací získáme pro klasickou metodu sdružených gradientů. Ostatní modifikace této metody již vedou ke zvýšení potřebného počtu operací. Je to způsobeno tím, že metoda sdružených gradientů konverguje po relativně malém počtu iterací a následná snížení počtu iterací v modifikovaných metodách nejsou tak velká, aby převážila zvýšení výpočetní složitosti jednotlivých iterací (kroky vykonané navíc). Metoda největšího spádu ve své základní podobě potřebuje velký počet operací, je to způsobeno velkým počtem iterací, které je nutno provést. Modifikace metody největšího spádu pak výrazně snižují počty iterací a zvýšení náročnosti výpočtu jednotlivých iterací není oproti klasické

metodě největšího spádu tak velké. Jediná metoda, která vykazuje výrazné zhoršení výpočetní složitosti metody největšího spádu, je metoda největšího spádu s oboustranným diagonálním předpodmíněním.

Dále bylo provedeno porovnání metod na základě měření času potřebného k výpočtu. Tento test by měl odpovídat vypočítanému počtu operací, neboť se v podstatě jedná o praktickou realizaci měření počtu operací. Pokud například nějaká metoda potřebuje poloviční počet operací než základní metoda, měla by počítat i dvakrát rychleji. Porovnání těchto dvou způsobů hodnocení jsou vidět z grafů na obrázcích (6.1) a (6.2). Pro každou metodu zobrazují počet operací a změřenou dobu výpočtu[1], první graf odpovídá výsledkům řešení soustavy 1, druhý graf výsledkům řešení soustavy 2.

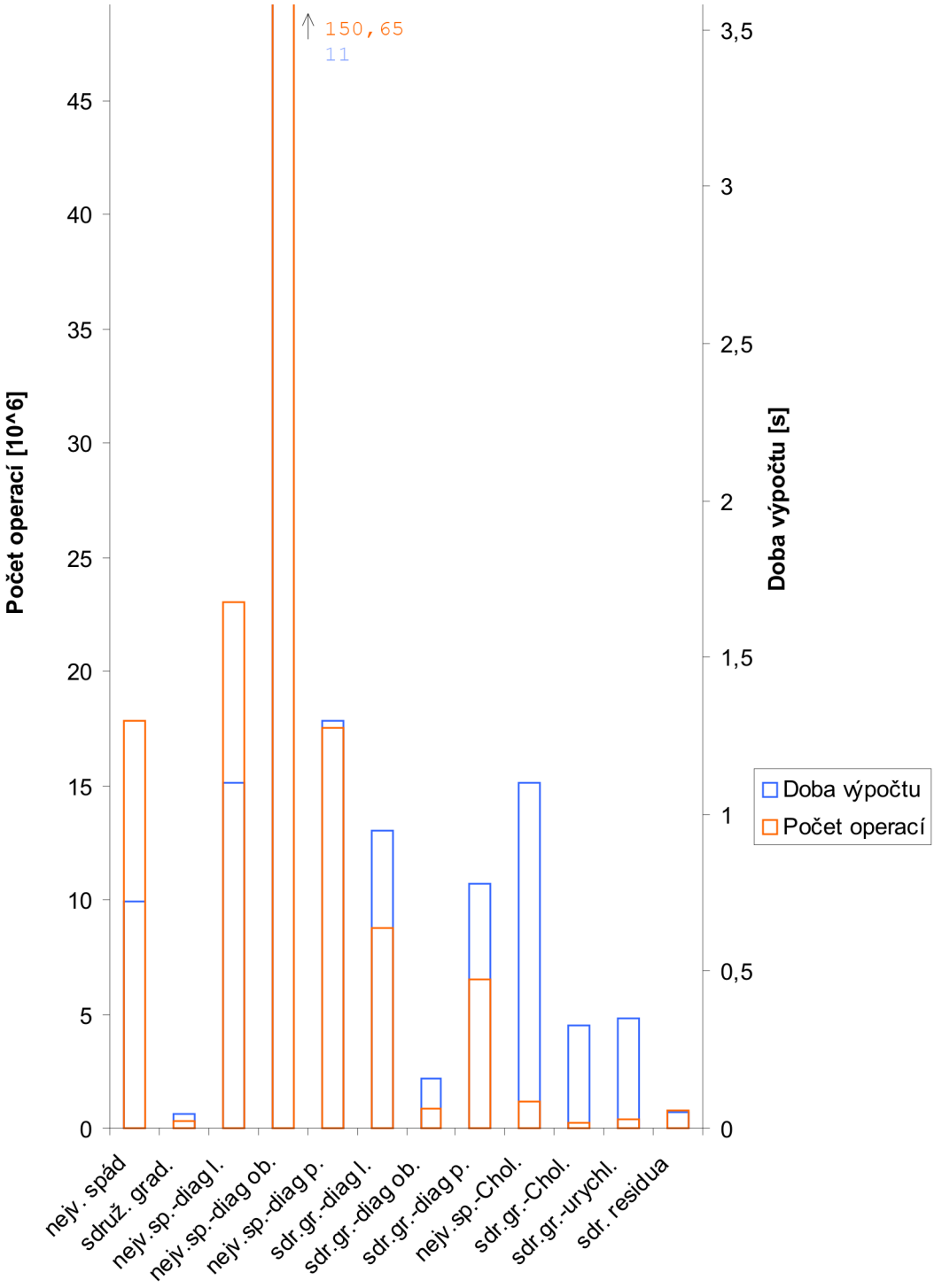
Porovnání počtu operací a změřené doby výpočtu v některých případech nekorespondují. Nejvýraznější rozpor je pro metody předpodmíněné neúplnou Choleského faktorizací a pro metodu sdružených gradientů s urychlovacím parametrem. V případě řešení soustavy 2 si počet operací a doba výpočtu přibližně odpovídají. V případě první soustavy však dostáváme zcela opačné výsledky. Teoreticky byl počet operací snížen nebo mírně snížen, ale praktická doba výpočtu se u všech tří metod zvýšila (až osminásobně).

Důvod pomalého výpočtu není jednoznačně identifikovatelný, vzhledem k tomu, že druhá soustava byla počítána přibližně stejnou dobu jako podle odhadu. Příčinou by mohlo být to, že u všech tří metod se využívá řídkost matice soustavy a soustava 2 má menší podíl nenulových prvků (8,8 %) než soustava 1 (15,9 %), tedy pracuje s méně prvky, na druhou stranu je vyššího řádu. Tento jev však uvažujeme i při konkrétním počítání počtu operací, proto by si čas a počet operací měly odpovídat.

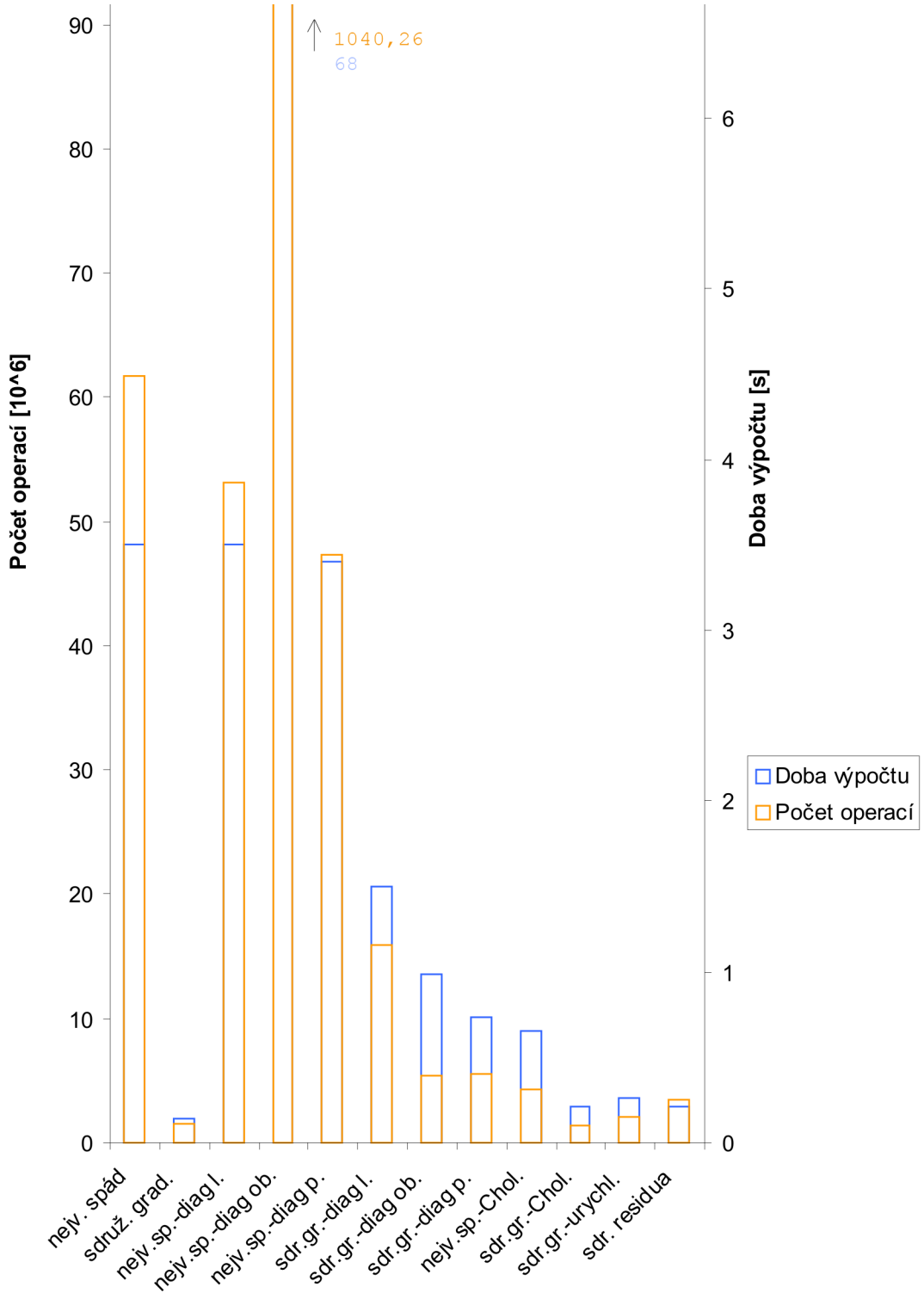
Mezi porovnáním počtu operací a dobou výpočtu dochází v případě některých

[1]Uvažujeme vždy čas měřený na prvním počítači – sloupec 1.

metod k rozporům. Oba způsoby porovnání mají své výhody a nevýhody a měli bychom je zohlednit oba (teoretické určení počtu operací je sice exaktní a jednoznačně popsatelné, ale při praktické realizaci náročnějšího problému je doba výpočtu rozhodujícím faktorem). Sledování počtu operací je výhodné v tom, že naprosto precizně spočítá počet operací, které je ve výpočtu nutno provést. Jedná se však pouze o teoretické stanovení výpočetní složitosti, nezahrnuje žádné jiné vlivy. Měření času má výhodu v tom, že zahrnuje všechny faktory, které dobu výpočtu ovlivňují. Tento přístup však vyžaduje dokonale optimalizovaný kód. Vzhledem k tomu, že získání zcela optimalizovaného kódu nebylo tématem této práce, musíme měření času brát spíše jako orientační. Pokud bychom se chtěli detailně zabývat samotným měřením času praktického výpočtu, bylo by nutné se samostatně věnovat pouze optimalizaci kódu, což by pojalo rozsah celé práce. Dalším problémem, na který naráží měření doby výpočtu, je kolísání naměřených hodnot časů.



Obrázek 6.1: Efektivita jednotlivých metod pro řešení soustavy 1



Obrázek 6.2: Efektivita jednotlivých metod pro řešení soustavy 2

Kapitola 7

Závěr

7.1 Shrnutí

V této práci jsem se zaměřila na řešení normálních rovnic, které se objevují ve vyrovnání geodetických měření. Podala jsem přehled klasických iteračních metod, které mohou sloužit k řešení normálních rovnic. Zaměřila jsem se na gradientní metody a na zkoumání různých možností modifikace obecně známých algoritmů. Jedním ze způsobů je předpodmínění, které by mělo sloužit ke zvýšení stability celého výpočtu, zároveň jsem se pokusila o co nejefektivnější implementaci, aby došlo i ke zrychlení konvergence výpočtu. Kromě předpodmínění jsem použila i jiný způsob modifikace, který rovněž zohledňuje vlastnosti matice normálních rovnic. Vlastní implementaci zkoumaných metod jsem provedla v programu Matlab®. Následné testování metod na reálných datech slouží k praktickému zhodnocení a porovnání jednotlivých metod. Je zvoleno více způsobů porovnání, abychom získali komplexní pohled na danou problematiku. Na základě provedené analýzy lze učinit následující závěry.

- Gradientní metody vykazují velmi dobré numerické vlastnosti. Zkoumala jsem metodu největšího spádu, která je přímočará a slouží spíše k ilustraci principu, na kterém jsou gradientní metody založeny. Rychlost konvergence není příliš veliká, několikanásobně převyšuje počet

neznámých n[1]. Pro praktické použití tedy není příliš vhodná. Některé modifikace mohou její numerické vlastnosti zlepšit. Vhodným způsobem je předpodmínění metodou neúplné Choleského faktorizace.

- Metoda sdružených gradientů má výborné numerické vlastnosti ve své standardně používané podobě. Je rychlá a stabilní, řešení s přiměřeně zvolenou přesností dosahuje po méně než n iteracích. Vzhledem k její propracovanosti a maximální efektivitě se těžko hledá vhodný způsob modifikace, který by její vlastnosti ještě zlepšil. Velice elegantní způsob je předpodmínění pomocí neúplné Choleského faktorizace. Tato metoda dokonce snižuje celkový potřebný počet opeací. Další variantou je předpodmínění s použitím urychlovacího parametru. Bylo by zajímavé podrobnější studium této metody, dostupná literatura použitá jako podklad pro tuto práci o ní více informací neuvádí. Bližší zkoumání předpodmínění s použitím urychlovacího parametru by mohl být jedním z bodů mé diplomové práce, kterou chci navázat na tuto bakalářskou práci. Další zajímavou metodou je metoda sdružených residuí, která poskytuje také velmi dobré výsledky.

- Modifikace klasických algoritmů má i své nevýhody, protože je většinou spojena s vykonáním některých operací navíc. Tyto nadbytečné kroky mohou mít negativní vliv na chování metody během výpočtu, dochází mnohdy k jeho zpomalení, ačkoliv byl snížen počet iterací. Některé způsoby předpomínění se na základě hodnotících kritérií ukázaly jako nevhodné. Jedná se o diagonální předpodmínění, které literatura [5] uvádí jako zvlášť vhodné pro symetrické a pozitivně definitní matice. V aplikaci na řešení normálních rovnic však nepodává uspokojivé výsledky.

[1]Bez uvažování zaokrouhlovacích chyb bychom u gradientních metod měli dosáhnout řešení po nejvýše n krocích.

7.2 Závěr

Předpodmínění a jiné další způsoby modifikace klasických metod mohou vést ke zvýšení stability a efektivity výpočtu normálních rovnic. Záleží na volbě způsobu předpodmínění. Špatně zvolená modifikace může vést dokonce ke zhoršení výpočetních vlastností metody. V této práci jsem se pokusila přiblížit některé z druhů předpodmínění a modifikací klasických algoritmů. Na základě praktických výpočtů jsem provedla jejich zhodnocení a analýzu.

Za přínos této práce považuji i její praktický výstup, jedná se o balík metod, které lze použít pro řešení soustav normálních rovnic. Všechny metody jsou k dispozici na přiloženém CD.

7.3 Výhled

Obsah této práce dává vzniknout řadě podnětů, kterými by bylo možné na tuto práci navázat. Některými bych se ráda zabývala ve své diplomové práci.

- Provést testování metod na rozsáhlejších soustavách normálních rovnic, sledovat chování jednotlivých metod se zvyšujícím se řádem soustavy.
- Více využít řídkost matice normálních rovnic při řešení soustavy.
- Zaměřit se na optimalizaci kódu.
- Provést implementaci metod v jiném prostředí (programovací jazyk C++), případně ve více různých prostředích a porovnat chování metod v závislosti na programovacím jazyku.

Literatura

[1] KABELÁČ, Josef. *Geodetické metody vyrovnání – Metoda nejmenších čtverců*. Plzeň: Západočeská univerzita, Fakulta aplikovaných věd, 2003. ISBN 80-7043-260-8.

[2] MÍKA, S. – BRANDNER, M. *Numerické metody I.* Plzeň: Západočeská univerzita, Fakulta aplikovaných věd, 2000. ISBN 80-7082-619-3.

[3] FILLER, Vratislav. *Varianty řešení soustav normálních rovnic*, [online]. Praha: České vysoké učení technické, Fakulta stavební, 2003. Dostupné z: ⟨ http://klobouk.fsv.cvut.cz/∼vrf/prace/mnc/mnc.html ⟩.

[4] GOLUB, Gene H. – VAN LOAN, Charles F. *Matrix Computations*. 3rd edition. Baltimore and London: The Johns Hopkins University Press, 1996. ISBN 0-8018-5414-8.

[5] QUARTERONI, A. – SACCO, R. – SALERI, F. *Numerical Mathematics*. 1st edition. Springer, 2000. ISBN 0-387-98959-5.

[6] AXELSSON, O. *Iterative Solution Methods*. Cambridge University Press, New Ed edition, March 29, 1996. ISBN 0-521-44524-8.

[7] STRANG, G. – BORRE, K. *Linear Algebra, Geodesy and GPS*. Wellesley Cambridge Press, 1997. ISBN 0-9614088-6-3.

[8] ROHN, Jiří. *Lineární algebra a optimalizace na slidech*, [online]. Praha: Ústav informatiky Akademie věd České republiky, Technical Report 905.

12. 2. 2004 [cit. 2005-10-06]. Dostupné z: ⟨http://www.cs.cas.cz/cgi-bin/extftp?ftp://ftp.cs.cas.cz/pub/reports/v905-04.ps⟩.

[9] KUTÁKOVÁ, H. – SMITKOVÁ, M. *Semestrální práce z předmětu Geodetické metody vyrovnání*. Plzeň: Západočeská univerzita, Fakulta aplikovaných věd, 2005.

[10] FELCMAN, Jiří. *Numerická matematika*, [online]. Praha: Univerzita Karlova, Matematicko-fyzikální fakulta, 20. 5. 2005 [cit. 2006-05-03]. Dostupné z: ⟨http://www.volny.cz/cpt.lee/pers/s_skola/nm-20_5_2005.pdf⟩.

[11] REKTORYS, K. a kol. *Přehled užité matematiky*. Prométheus Praha, 6., přepracované vydání, 1995. ISBN 80-85849-72-0.

[12] HOLÁ, Eva. *Užití metody regularizace v geodézii*. Diplomová práce, Praha: České vysoké učení technické, Fakulta stavební, 1983.

[13] *Konzultace s vedoucím bakalářské práce Ing. Vratislavem Fillerem, Ph.D.*

[14] *Nápověda k programu Matlab® verze 7.0 (R14)*. The MathWorks, Inc. 2006. Dostupné z: ⟨http://www.mathworks.com/access/helpdesk/help/techdoc/matlab.shtml⟩.

[15] KADLEC, M. *Porovnání různých způsobů vyrovnání v geoetické praxi*. Diplomová práce, Plzeň: Západočeská univerzita, Fakulta aplikovaných věd, 2005.

[16] DANĚK, J. *Soustavy lineárních algebraických rovnic*. Plzeň: Západočeská univerzita, Fakulta aplikovaných věd, [cit. 2006-05-10]. Dostupné z: ⟨http://www.cam.zcu.cz/∼danek/Students/2003_ZS/Materialy/soustavy_linearnich_rovnic.pdf⟩.

Dodatek A

Přílohy

Metoda největšího spádu s předpodmíněním neúplnou Choleského faktorizací

```
function[x,k] = SD_chol(A,b,tol);
n=size(A,1);
% neuplny Choleskeho rozklad matice A
pom=A;
for k=1:n
    pom(k,k)=sqrt(pom(k,k));
    for i=k+1:n
        if (pom(i,k)~=0)
            pom(i,k)=pom(i,k)/pom(k,k);
        end
    end
    for j=k+1:n
        for i=j:n
            if pom(i,j)~=0
                pom(i,j)=pom(i,j)-pom(i,k)*pom(j,k);
            end
        end
    end
```

```
end

H=zeros(n);
for i=1:n
    for j=1:n
        if i>=j
            H(i,j)=pom(i,j);
        end
    end
end

M=H*H'; % matice predpodmineni M

% inicializace vektoru reseni
x=zeros(n,1);
% pocatecni hodnota residua
r=b-A*x;
% pocitadlo iteraci
k=0;

% vlastni reseni soustavy rovnic
while (sqrt(r'*r) > tol) % volba zastav. podminky (norma residua)
    % reseni pomocne soustavy M*z=r (M=H*H') metodou cholesky
    z=cholesky(H,r);
    alpha=r'*z/(z'*A*z);
    x=x+alpha*z;
    r=b-A*x;
    k=k+1;
end
```

Metoda sdružených gradientů s předpodmíněním neúplnou Choleského faktorizací

```
function[x,k]= CG_chol(A,b,tol);
n=size(A,1);
%inicializace vektoru reseni
x=zeros(n,1);
%pocatecni hodnota residua
r_0=b-A*x;

%neuplny Choleskeho rozklad matice A
pom=A;
for k=1:n
    pom(k,k)=sqrt(pom(k,k));
    for i=k+1:n
        if (pom(i,k)~=0)
            pom(i,k)=pom(i,k)/pom(k,k);
        end
    end
    for j=k+1:n
        for i=j:n
            if pom(i,j)~=0
                pom(i,j)=pom(i,j)-pom(i,k)*pom(j,k);
            end
        end
    end
end

H=zeros(n);
for i=1:n
    for j=1:n
        if i>=j
```

```
            H(i,j)=pom(i,j);
        end
    end
end
H;
M=H*H'; %matice predpodmineni M

%vlastni reseni soustavy rovnic
%prvni iterace
%reseni pomocne soustavy rovnic M*z_0=r_0 (M=H*H') metodou cholesky
z_0=cholesky(H,r_0);

p=z_0;
alpha=r_0'*z_0/(p'*A*p);
x=x+alpha*p;
r=r_0-alpha*A*p;
%pocitadlo iteraci
k=1;

%vektory r_i, r_j, z_j uchovavaji hodnoty residui ze dvou po sobe
%jdoucich iteraci k-1, k-2
r_j=r_0;
r_i=r;
z_j=z_0;

while (sqrt(r_i'*r_i) > tol) %volba zastav. podminky (norma residua)
    %reseni pomocne M*z_i=r_i (M=H*H') metodou cholesky
    z_i=cholesky(H,r_i);
    beta=r_i'*z_i/(r_j'*z_j);
    p=z_i+beta*p;
    Ap=A*p;
```

```
    alpha=r_i'*z_i/(p'*Ap);
    x=x+alpha*p;
    r_j=r_i;
    r_i=r_i-alpha*Ap;
    z_j=z_i;
    k=k+1;
end
```

Metoda sdružených residuí

```
function [x,k]=conjugateResiduals(A,b,tol);
n=size(A,1);
%inicializace vektoru reseni
x=zeros(n,1);
%pocitadlo iteraci
k=0;
%pocatecni hodnota residua
r_i=b-A*x;

while (sqrt(r_i'*r_i) > tol) %volba zastav. podminky (norma residua)
    k=k+1;
    if k==1
        p_i=r_i;
        Api=A*p_i;
        riAri=r_i'*A*r_i;
    else
        riAri=r_i'*A*r_i;
        beta=(riAri)/(r_j'*A*r_j);
        p_i=r_i+beta*p_j;
        Api=A*p_i;
    end
    alpha=(riAri)/(Api'*Api);
```

```
    x=x+alpha*p_i;
    r=r_i-alpha*Api;
    r_j=r_i;
    r_i=r;
    p_j=p_i;
end
```

Metoda sdružených gradientů s diagonálním předpodmíněním zleva

```
function[x,k]=diagonalCG_l(A,b,tol);
n=size(A,1);
%vypocet matice predpodmineni C
C=diag([diag(A)]);
%vypocet inverze matice C volanim funkce inverse
C_inv=inverse(C);

%modifikace vstupni soustavy Ax=b na tvar C_inv*A*x=C_inv*b
%vypocet AA=C_inv*A;
for i=1:n
    AA(i,1:n)=C_inv(i,i)*A(i,1:n);
end

%vypocet bb=C_inv*b;
for i=1:n
    bb(i,1)=C_inv(i,i)*b(i,1);
end

%inicializace vektoru reseni
x=zeros(n,1);
%pocitadlo iteraci
k=0;
%pocatecni hodnota residua
```

```
r_x=bb-AA*x;
r_i=r_x;
while (sqrt(r_x'*r_x) > tol) %volba zastav. podminky (norma residua)
    k=k+1;
    if k==1
        p_i=r_i;
    else
        beta=r_i'*r_i/(r_j'*r_j);
        p_i=r_i+beta*p_j;
    end
    AApi=AA*p_i;
    alpha=r_i'*r_i/(p_i'*AApi);
    x=x+alpha*p_i;
    r=r_i-alpha*AApi;
    r_j=r_i;
    r_i=r;
    p_j=p_i;
    %dopocitani residua r_x puvodni soustavy, kvuli sledovani
    %velikosti residua pro zastaveni vypoctu
    r_x=b-A*x;
end
```

Dodatek B

Obsah přiloženého CD

Přiložené CD obsahuje následující adresáře:

- Adresář `metody` obsahuje všechny metody naprogramované a použité v této práci.
- Adreář `data_a_spusteni` obsahuje testovaci soustavy a příslušné soubory pro spuštění řešení soustav všemi metodami v programu Matlab®.
- Adresář `dokumentace` obsahuje soubor .pdf s vlastním textem práce.

Evidenční list

Souhlasím s tím, aby moje bakalářská práce byla půjčována k prezenčnímu studiu v Univerzitní knihovně ZČU v Plzni.

V Plzni dne 12. června 2006 .....
podpis

Uživatel stvrzuje svým čitelným podpisem, že tuto bakalářskou práci použil ke studijním účelům a prohlašuje, že ji uvede mezi použitými prameny.

Jméno	**Fakulta / katedra**	**Datum**	**Podpis**