VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra aplikované matematiky

Algoritmy kvadratického programování a jejich aplikace

Quadratic programming algorithms and their application

2018

Jan Přindiš

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra aplikované matematiky

Zadání bakalářské práce

Student: **Jan Přindiš**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 1103R031 Výpočetní matematika

Téma: Algoritmy kvadratického programování a jejich aplikace
Quadratic programming algorithms and their application

Jazyk vypracování: čeština

Zásady pro vypracování:

Cílem práce je seznámit se s efektivními algoritmy kvadratického programování, jejich implementací, s jejich použitím ve vybraných oblastech aplikací (support vector machines, mechanika, optimální řízení) a podílet se na jejich vývoji.

Seznam doporučené odborné literatury:

Z. Dostál, Optimal quadratic programming algorithms, Springer, New York 2010,
Z. Dostál, P. Beremlijský, Metody optimalizace, Ostrava 2013

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **prof. RNDr. Zdeněk Dostál, DSc.**

Datum zadání: 01.09.2017
Datum odevzdání: 30.04.2018

VYSOKÁ ŠKOLA BÁŇSKÁ-TECHNICKÁ UNIVERZITA Fakulta elektrotechniky a informatiky 1D16 OSTRAVA

doc. RNDr. Jiří Bouchala, Ph.D.
vedoucí katedry

prof. Ing. Pavel Brandštetter, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 30. dubna 2018

.....

Souhlasím se zveřejněním této bakalářské práce dle požadavků čl. 26, odst. 9 Studijního a zkušebního řádu pro studium v bakalářských programech VŠB-TU Ostrava.

V Ostravě 30. dubna 2018 .....

Na tomto místě bych rád poděkoval prof. RNDr. Z. Dostálovi, DSc. za jeho cenné rady a trpělivost při vedení mé bakalářské práce.

Abstrakt

Tato práce se zabývá algoritmy k řešení úloh kvadratického programování bez omezení, konkrétně se jedná o gradientní metody, kterými jsou metoda největšího spádu, metoda Barzilai-Borwein a metoda sdružených gradientů. Věnovat se bude i předpodmíněním těchto metod. Cílem této práce je seznámit se s algoritmy kvadratického programování, jejich implementací a také aplikacemi.

Klíčová slova: Optimalizační úloha, Kvadratické programování bez omezení, Metoda největšího spádu, Metoda Barzilai-Borwein, Metoda sdružených gradientů, Metoda předpodmíněných sdružených gradientů, Předpodmínění SSOR

Abstract

This thesis deals with algorithms for solving unconstrained quadratic programming problems, in particular is about gradient methods as steepest descent method, Barzilai-Borwein algorithm and conjugate gradient method and preconditioning of these mothods. The goal of this thesis is to deal with algorithms of quadratic programming, their implementation and also their application.

Key Words: Optimization problem, Unconstrained quadratic programmnig, Method of steepest descent, Barzilai-Borwein method, Conjugate gradient method, Preconditioned conjugate gradients method, SSOR precondition

Seznam použitých zkratek a symbolů

α, ϵ	–	Skaláry nebo skalární funkce
x, g	–	Vektory
A, M	–	Matice
$\sigma(A)$	–	Spektrum matice A
$\kappa(A)$	–	Spektrální číslo podmíněnosti matice A
$\mathrm{Span}\{x_1, \ldots, x_k\}$	–	Lineární obal
f, g	–	Zobrazení z $\mathbb{R}^n$ do $\mathbb{R}$
Ω	–	Množina
$\|\|x\|\|$	–	Euklidovská norma vektoru x
(x, v)	–	Euklidovský skalární součin vektorů x a y
k	–	Čítač
QP	–	Kvadratické programování (Quadratic programming)
BB	–	Metoda Barzilai-Borwein
CG	–	Metoda sdružených gradientů (Conjugate gradient method)
SD	–	Metoda největšího spádu (Method of steepest descent)
PBB	–	Předpodmíněná metoda Barzilai-Borwein
PCG	–	Předpodmíněná metoda sdružených gradientů (Preconditioned conjugate gradients method)
PSD	–	Předpodmíněná metoda největšího spádu (Preconditioned method of steepest descent)
SPD	–	Symetrická a pozitivně definitní (Symetric and positive definite)
SSOR	–	Předpodmínění SSOR (Symmetric successive over-relaxation)

Obsah

Seznam obrázků

Seznam algoritmů

1 Úvod

Obor optimalizace si klade za cíl nalézt nejlépe podmíněný prvek či metodu na zkoumané množině, respektive procesu. Z matematického hlediska se jedná o maximalizaci či minimalizaci dané funkce, kterou nazýváme funkcí cenovou. Z fyzikálního hlediska může mít řešení minimalizace cenové funkce význam energie systému v rovnovážném stavu.

Za předpokladu, že předpis cenové funkce je kvadratický, nazýváme naši úlohu úlohou kvadratického programování (QP), na které v praxi vedou různé teoretické i praktické úlohy. Z konkrétních oblastí můžeme zmínit úlohy kontaktních problémů, optimálního řízení, úpravy obrazu, strojového učení nebo umělé inteligence.

Základním dělením úloh QP je na úlohy s omezením a bez omezení, tímto se nám blíže specifikuje množina, na které budeme řešení hledat. V této práci budou popsány úlohy QP bez omezení a konkrétně pro minimalizace. Algoritmy k řešení úloh QP s omezením jsou však často založeny na obdobných principech, liší se úpravami pro odlišné vstupní podmínky.

Další dělení úloh QP může být podle jejich zadání ve smyslu velikosti úlohy, jinak samozřejmě budeme přistupovat k úlohám o jednotkách proměnných než k úlohám o milionu a více proměnných, kde se v dnešní době běžně užívá různých předpodmínění, rozkladů a diskretizací pro dosažení lepších podmínek vypočítatelnosti. Kritériem pro algoritmy řešící tyto úlohy bude jednoznačně jejich rychlost nalezení řešení nebo také rychlost konvergence, avšak dalšími důležitými aspekty bude jejich výpočetní složitost, jejich nároky na paměť, náročnost implementace nebo také přesnost dosaženého řešení v určitém časovém horizontu.

V této práci se budeme věnovat třem metodám k řešení výše specifikovaných úloh, kterými jsou metoda největšího spádu (SD), metoda Borzilai-Borwein (BB) a také velmi používaná metoda sdružených gradientů (CG). První uvedená metoda SD je příkladem velmi jednoduchého a robustního algoritmu, který však není vhodný pro velké nebo špatně podmíněné úlohy. Metoda BB je také poměrně jednoduchá a robustní, ale dosahuje výrazně lepších rychlostí konvergence a při určité adaptaci je vhodné ji použít i pro řešení velkých úloh, avšak ve většině případů je nejefektivnější poslední uvedená metoda CG, která se zakládá na sdružených směrech. Uvedeme si předpodmínění výše zmíněných metod, které nám mohou za určitých podmínek zlepšit konvergence vybraných metod.

V závěru práce se budeme věnovat numerickým experimentům pro porovnání všech popsaných metod na modelových úlohách, které byly také motivací pro vytvoření této práce. K porovnání metod si vytvoříme tři úlohy, první z nich bude dvoudimenzionální, abychom mohli sledovat konvergenci řešení na vrstevnicích funkce, následovat bude úloha se vstupní malou, řídkou a dobře podmíněnou maticí a poslední úloha bude metody porovnávat na řídké, avšak špatně podmíněné matici.

K implementaci algoritmů a řešení úloh byl použit software *MatLab*.

2 Úlohy kvadratického programování

Úlohy kvadratického programování (QP) jsou popsány kvadratickou cenovou funkcí f

$$f(x) = \frac{1}{2}x^T Ax - b^T x \tag{2.0.1}$$

definovanou na množině $D \subseteq \mathbb{R}^n$, kde $A \in \mathbb{R}^{n\times n}$ je symetrickou maticí ($A = A^T$) řádu n a $b \in \mathbb{R}^n$ je vektorem pravých stran. Gradientem funkce f pro dané x je

$$\nabla f(x) = Ax - b \tag{2.0.2}$$

a hessiánem pak je

$$\nabla^2 f(x) = A.$$

Vektor $d \in \mathbb{R}^n$ budeme nazývat vektorem poklesu za předpokladu, že pro každé dostatečně malé nenulové $\epsilon \in \mathbb{R}$ bude platit, že

$$f(x + \epsilon d) < f(x).$$

Z Taylorova rozvoje je možné odvodit také

$$f(x + d) = f(x) + (Ax - b)^T d + \tfrac{1}{2}d^T Ad.$$

stejně jako

$$f(x + \epsilon d) = f(x) + \epsilon(Ax - b)^T d + \frac{\epsilon^2}{2}d^T Ad. \tag{2.0.3}$$

Pak tedy platí, že d bude vektorem poklesu jen tehdy, když

$$(Ax - b)^T d < 0.$$

2.1 Minimalizační úlohy kvadratického programování

Řešením minimalizace funkce f (2.0.1) je takové $\bar{x} \in D$, které splňuje následující podmínku

$$f(\bar{x}) \leq f(x), x \in \mathbb{R}^n.$$

V případě hledání řešení minimalizační úlohy s omezením by řešení $\bar{x}$ pocházelo z množiny $\Omega \subseteq D$, která by byla popsána lineárními rovnostmi a nerovnostmi. Řešení by splňovalo, že

$$f(\bar{x}) \leq f(x), x \in \Omega.$$

Tuto minimalizaci funkce f můžeme značit také jako

$$\min_{x\in D} f(x), \tag{2.1.1}$$

respektive

$$\min_{x\in \Omega} f(x).$$

2.2 Podmínky minima úlohy kvadratického programování bez omezení

Věta 1 *Vektor $\bar{x}$ bude řešením QP úlohy bez omezení* (2.1.1) *pouze a jen tehdy, když matice A bude pozitivně semidefinitní a zároveň bude platit, že*

$$\nabla f(x) = Ax - b = o. \tag{2.2.1}$$

Důkaz Pokud $\bar{x}$ a $d \in \mathbb{R}^n$, $\epsilon \in \mathbb{R}$, pak můžeme z rovnice (2.0.3) usoudit, že

$$f(\bar{x} + \epsilon d) - f(\bar{x}) = \epsilon(Ax - b)^T d + \frac{\epsilon^2}{2} d^T A d. \tag{2.2.2}$$

a pokud je $\bar{x}$ řešením, tak je pravá strana rovnice (2.2.2) nezáporná pro libovolná ϵ i d. Pro dostatečně velké ϵ bude z nezápornosti pravé strany vyplývat $d^T A d > 0$, tedy A bude pozitivně semidefinitní. V případě dostatečně malých ϵ bude z nezápornosti pravé strany vyplývat $(A\bar{x} - b)^T d = 0$ pro libovolné $d \in \mathbb{R}^n$. Pak tedy $Ax - b = o$. Z toho nám plyne, že pro semidefinitní matici A je $\bar{x}$ řešením (2.1.1) a pro libovolné d platí

$$f(\bar{x} + d) - f(\bar{x}) = \tfrac{1}{2} d^T A d \leq 0.$$

■

Věta 2 *Vektor $\bar{x}$ bude jediným řešením úlohy bez omezení* (2.1.1) *pouze a jen tehdy, když matice A bude pozitivně definitní.*

Důkaz Pokud $\bar{x}$ je jediným řešením (2.1.1) a A je podle (2.2) pozitivně semidefinitní, stejně jako je $\bar{x}$ jediným vektorem, který splňuje $A\bar{x} = b$, tedy A je regulární a zároveň pozitivně semidefinitní, plyne z toho, že A je pozitivně definitní. Z pozitivní defintnosti A a z (2.2.1) zase plyne jedinečnost řešení. ■

Věta 3 *Nechť matice A bude pozitivně definitní, řešení soustavy $Ax = b$ bude ekvivalentní s řešením minimalizační úlohy* (2.1.1).

Důkaz Budeme-li předpokládat, že $x + \alpha d$ je přírůstkem bodu x, kde $x, v \in \mathbb{R}^n$ a $\alpha \in \mathbb{R}$, pak rozdíl přírůstku od bodu x můžeme vyjádřit jako

$$\begin{aligned} f(x + \alpha v) - f(x) &= \frac{1}{2}(A(x + \alpha v), x + \alpha v) - (b, x + \alpha v) - \frac{1}{2}(Ax, x) - (b, x) \\ &= \alpha(Ax, v) - \alpha(b, v) + \frac{1}{2}\alpha^2(Av, v) \\ &= \frac{1}{2}\alpha^2(Av, v) + \alpha(Ax - b, v). \end{aligned}$$

Budeme dále předpokládat, že $\bar{x}$ je řešením soustavy $A\bar{x} = b$ a tudíž i $A\bar{x} - b = o$, což splňuje nutnou podmínku minima existence nulové parciální derivace , tj. $\nabla f(\bar{x}) = 0$ a zároveň $\nabla f(\bar{x}) = A\bar{x} - b$. Z pozitivní definitnosti můžeme odvodit, že

$$f(\bar{x} + \alpha v) - f(\bar{x}) = \tfrac{1}{2}\alpha^2(Av, v) \geq 0, \qquad \forall \alpha \in \mathbb{R}, \forall v \in \mathbb{R}^n.$$

Což můžeme přepsat také jako

$$f(\bar{x} + \alpha v) \geq f(\bar{x}), \qquad \forall \alpha \in \mathbb{R}, \forall v \in \mathbb{R}^n,$$

přičemž z této nerovnosti plyne, že v libovolném směru v z bodu $\bar{x}$ funkční hodnota roste nebo nabývá stejných hodnot, pak tedy můžeme nazvat $\bar{x}$ minimem funkce f a zároveň řešením soustavy $A\bar{x} = b$.

■

3 Gradientní metody

Obecným pravidlem pro gradientní metody je, že k dosažení řešení úlohy QP kombinují aproximace s iteracemi, díky čemuž je dosaženo postupně dostatečně přesného odhadu řešení minimalizační úlohy. Proměnná x v příslušné iteraci bude značena

$$x_k$$

a pro lineárního model gradientních metod bude platit, že

$$x_{k+1} = x_k + \alpha_k d_k, \tag{3.0.1}$$

kde $\alpha_k \in \mathbb{R}$ je délkou kroku v dané iteraci a $d_k \in \mathbb{R}^n$ je směr vektoru poklesu v dané iteraci, který je závislý na směru gradientu nebo je jím gradient samotný, z čehož je odvozen název těchto metod.

Ještě můžeme doplnit, že pro metody založené na lineárním modelu platí, že mají nízkou cenu iterace a jsou robustní, tedy jsou aplikovatelné na širší spektrum úloh, což má ale za následek velký počet iterací algoritmu k dosažení požadované přesnosti řešení. Tento fakt však není překvapením, jelikož nepracují např. oproti metodě sdružených gradientů s informacemi z předešlých iterací.

Volba směru poklesu

Uvedeme si směr, ve kterém funkce nejvíce klesá, tedy směr největšího spádu. Odvodíme ho z derivace funkce f ve směru v_k

$$\frac{\mathrm{d}f(x_k)}{\mathrm{d}v_k} = (\nabla f(x_k), v_k) = \cos(\varphi)||g_k||||v_k||, \tag{3.0.2}$$

kde hodnota derivace bude nejmenší, pokud $\cos\varphi = -1 \Leftrightarrow \varphi = \pi$. Pokud tedy vektory v_k a g_k svírají úhel π, pak $v_k = -g_k$.

Volba délky kroku

Jeden z přístupů, jak definovat délku kroku je volba tzv. optimálního kroku α_{opt}. Takováto délka kroku minimalizuje hodnotu cenové funkce f v daném směru poklesu d. To si můžeme odvodit z následující derivace

$$\frac{\mathrm{d}f(x-\alpha d)}{\mathrm{d}\alpha} = \alpha d^T A d - b^T d - x^T A d = \alpha d^T A d - d^T(Ax - b) = \alpha d^T A d - d^T g,$$

z podmínky minima pak vychází,že

$$\alpha_{\mathrm{opt}} d^T A d - d^T g = 0 \Leftrightarrow \alpha_{\mathrm{opt}} = d^T g / d^T A d. \tag{3.0.3}$$

Problém takovéto délky kroku je záruka pouze lokálního poklesu, avšak globální pokles nemusí být výrazný při špatné podmíněnosti matice A nebo její dostatečně velké dimenzi, algoritmus pak musí iterovat krok až příliš mnohokrát a reálné ukončení algoritmu při dosažení požadované přesnosti nemusí nastat v rozumném čase.

Pro krok α_{opt} v uvedeném směru d platí, že minimalizuje $f(x_k - \alpha d)$ pro $\alpha > 0$, pro takovéto α a řešení $\bar{x}$ platí

$$f(x_k - \alpha_{\text{opt}} d) - f(x) \leq f(x_k - \alpha d) - f(\bar{x}).$$

Budeme-li zkoumat energetickou normu pro $x \in \mathbb{R}^n$, kterou můžeme vyjádřit jako

$$||x||_A^2 = x^T A x,$$

potom

$$||x - \bar{x}||_A^2 = (x - \bar{x})^T A (x - \bar{x}) = (x - A^{-1}b)^T A (x - A^{-1}b) = 2(f(x) - f(\bar{x})),$$

a pro libovolné $\alpha > 0$

$$||x_{k+1} - \bar{x}||_A^2 \leq ||x_k - \alpha g_k - \bar{x}||_A^2.$$

Z tohoto vyjádření můžeme usoudit, že α_{opt} neminimalizuje vzdálenost od řešení.

Jiným přístupem k volbě délky kroku pro gradientní metodu může být volba pevného (konstantního) kroku $\bar{\alpha}$, pak platí, že pro libovolnou iteraci k je $\alpha_k = \bar{\alpha}$ a

$$x_{k+1} = x_k + \bar{\alpha} d_k.$$

Z vlastností normy a soustavy $A\bar{x} = b$, kde $\bar{x}$ je řešením soustavy můžeme odvodit

$$\begin{aligned} ||x_{k+1} - \bar{x}|| = ||x_k - \bar{\alpha} g_k - \bar{x}|| = ||x_k - \bar{\alpha}(Ax_k - b) - \bar{x} - \bar{\alpha}((A\bar{x} - b))|| = \\ = ||(I - \bar{\alpha}A)(x_k - \bar{x})|| \leq ||(I - \bar{\alpha}A)||||(x_k - \bar{x})|| = \\ = \max_{\lambda_i \in \sigma(A)} |1 - \bar{\alpha}\lambda_i| ||(x_k - \bar{x})|| = \max\{1 - \bar{\alpha}\lambda_{\min}, \bar{\alpha}\lambda_{\max} - 1\} ||(x_k - \bar{x})||, \end{aligned}$$

kde λ_{min} a λ_{max} jsou nejmenším a největším vlastním číslem matice A a $\sigma(A)$ je jejím spektrem. Lze pak odvodit, že absolutní hodnota výrazu je menší než jedna pro

$$\bar{\alpha} \in (0, 2\lambda_{\max}^{-1})$$

Nejmenší hodnoty pak bude nabývat, když

$$1 - \bar{\alpha}\lambda_{\min} = \bar{\alpha}\lambda_{\max} - 1,$$

odkud odvodíme, že délka kroku $\bar{\alpha}_{\text{opt}}$ má předpis

$$\bar{\alpha}_{\text{opt}} = 2/(\lambda_{\min} + \lambda_{\max}).$$

Poznámka 1 Konkrétně pro metodu největšího spádu, která bude blíže popsána v podkapitole 3.1, ještě můžeme zmínit volbu délky kroku α_k jako

$$\alpha_k = \frac{1}{||A||}.$$

Tato volba kroku se nazývá Richardsonovou metodou, která souvisí s odhadem posloupnosti koeficientů pomocí vlastních čísel a samotné koeficienty se nazývají Rayleighovými podíly.

Odlišné přístupy volby délky kroku budou uvedeny u konkrétních vybraných algoritmů.

Ukončovací podmínky

Pro ukončovací podmínku algoritmu se nabízí několik variant, jedna z intuitivních podmínek je změna funkční hodnoty v posledním kroce, tedy $||f(x_k) - f(x_{k-1})||$. Alternativou pak může být délka posledního kroku $||x_k - x_{k-1}||$ nebo jen velikost gradientu $||g_k||$.

Avšak pro úlohy o větší dimenzi n matice A je složité očekávat exaktní přesnost řešení, takže vhodnější je požadovat slabší podmínku pro dostatečně velkou přesnost

$$||g_k|| \leq \epsilon,$$

kde $\epsilon \in \mathbb{R}$ je dostatečně malé a zároveň $\epsilon > 0$. Obdobně vhodnou podmínkou, která byla také použita při implementaci algoritmů v rámci této práce, je podmínka relativní velikosti gradientu, tj.

$$||g_k||/||g_{k_0}|| \leq \epsilon.$$

Algoritmus gradientní metody s optimální pevnou délkou kroku

Uvedeme si příklad algoritmu s výše zmíněnou volbou optimální pevné délky kroku $\bar{\alpha}_{opt} = 2/(\lambda_{\min} + \lambda_{\max})$, směrem poklesu zvolíme $-g_k$.

Je dána SPD matice $A \in \mathbb{R}^{n\times n}$, její nejmenší a největší vlastní čísla λ_{min} a λ_{max}, vektor $b \in \mathbb{R}^n$, dostatečně malou přesnost $\epsilon \in \mathbb{R} \wedge \epsilon > 0$ a $x_0 \in \mathbb{R}^n$.

input: $A, b, x_0, \epsilon, \lambda_{min}, \lambda_{max}$

$g_0 = Ax_0 - b, k = 0, \bar{\alpha}_{opt} = 2/(\lambda_{\min} + \lambda_{\max})$

while $||g_k||/||g_{k_0}|| \leq \epsilon$ **do**

$x_{k+1} = x_k - \bar{\alpha}_{opt} g_k$

$g_{k+1} = Ax_{k+1} - b$

$k = k + 1$

end

$\bar{x} = x_k$

Algoritmus 1: Gradientní metoda s optimální pevnou délkou kroku

Konvergence tohoto algoritmu při dané pevné délce kroku $\bar{\alpha}_{opt}$ je

$$||e_{k+1}|| \leq 1 - \frac{2\lambda_{\min}}{(\lambda_{\min}+\lambda_{\max})}||e_k|| = \frac{\kappa(A)-1}{\kappa(A)+1}||e_k||,$$

kde $e_k = x_k - \bar{x}$ a $\kappa(A)$ je spektrální číslo podmíněnosti matice A, které je definováno jako

$$\kappa(A) = \lambda_{\min}/\lambda_{\max}.[3]$$

3.1 Metoda největšího spádu

Metoda největšího spádu (SD) vychází z již popsaných přístupů, vychází z předpisu 3.0.1, tedy

$$x_{k+1} = x_k + \alpha_k d_k,$$

kde směrem poklesu d_k bude směr největšího spádu, kterým je záporný směr gradientu $-g_k$, délkou kroku v dané iteraci je α_{opt} 3.0.3 lokálně minimalizující funkci f 2.0.1

$$\alpha_{\text{opt}} = g_k^T d_k / d_k^T A d_k.$$

Po dosazení směru poklesu $-g_k$ do vzorce délky kroku v dané iteraci dostaneme

$$\alpha_k = -g_k^T g_k / g_k^T A g_k.$$

Dosazením do lineárního modelu dostaneme

$$x_{k+1} = x_k + \alpha_k d_k = x_k - \alpha_k g_k = x_k + \frac{g_k^T g_k}{g_k^T A g_k} g_k.$$

Je dána SPD matice $A \in \mathbb{R}^{n\times n}$, vektor $b \in \mathbb{R}^n$, dostatečně malou přesnost $\epsilon \in \mathbb{R} \wedge \epsilon > 0$ a $x_0 \in \mathbb{R}^n$.
input: A, b, x_0, ϵ
$g_0 = Ax_0 - b, k = 0$
while $||g_k||/||g_{k_0}|| \leq \epsilon$ **do**
$\alpha_k = -g_k^T g_k / g_k^T A g_k$
$x_{k+1} = x_k - \alpha_k g_k$
$g_{k+1} = Ax_{k+1} - b$
$k = k + 1$
end
$\bar{x} = x_k$

Algoritmus 2: Metoda největšího spádu

Poznámka 2 Iteraci gradientu g_{k+1} je pro implementaci možné definovat jako

$$g_{k+1} = Ax_{k+1} - b = A(x_k - \alpha_k g_k) - b = Ax_k + \alpha_k A g_k - b = g_k + \alpha_k A g_k,$$

využijeme tak spočteného Ag_k z výpočtu délky kroku a nemusíme tak násobit Ax_k.

Metoda SD konverguje pro libovolné počáteční $x_0 \in \mathbb{R}^n$ lineárně a rychlostí závislou na čísle podmíněnosti matice A, z čehož plyne, že při špatné podmíněnosti je tato metoda velmi pomalá. Pro odhad konvergence platí

$$||e_k||_A \leq (\frac{\kappa(A)-1}{\kappa(A)+1})_k||e_0||_A,$$

kde $e_k = x_k - \bar{x}$ představuje chybu v dané iteraci.[4]

Poznámka 3 Metoda SD vyhledává pomocí vlastních vektorů a to odpovídajícím od největších po nejmenší vlastní čísla matice A s normovaným gradientem blížícím se střídavě dvěma směrům p a p', které definujeme jako

$$p = \lim_{k\to\infty} \frac{g_{2k}}{||g_{2k}||} \text{ a } p' = \lim_{k\to\infty} \frac{g_{2k+1}}{||g_{2k+1}||},$$

a od kterých se odvíjí typický střídavě cyklický pokles.[8]

3.2 Metoda Barzilai-Borwein

Metoda Barzilai-Borwein vychází z metody SD, zakládá se obdobném lineárním modelu a užívá i stejného směru poklesu, avšak liší se délkou kroku v dané iteraci. Pravděpodobnou motivací pro vznik této metody bylo, krom snahy o překonání rychlosti konvergence metody SD, zakomponování informace z předešlých kroků algoritmu a odstranění cyklického chování metody SD právě pomocí vhodnější volby délky kroku, která vychází z přepisu

$$x_{k+1} = x_k - D_k g_k,$$

kde $D_k = \alpha_k I$. Pak k dosažení Quasi-Newtonovy vlastnosti hledáme α_k jako

$$\alpha_k = \text{argmin}_\alpha ||s_{k-1} - D_k y_{k-1}||.$$

Po úpravách dojdeme k tomu, že

$$\alpha_k = \frac{(s_{k-1}^T y_{k-1})}{(y_{k-1}^T y_{k-1})},$$

kde

$$s_{k-1} = x_k - x_{k-1}, \quad y_{k-1} = g_k - g_{k-1}.$$

Symetricky můžeme minimalizovat $\alpha_k = \text{argmin}_\alpha ||D_k^{-1} s_{k-1} - y_{k-1}||$, odkud vyjádříme α_k jako

$$\alpha_k = \frac{(s_{k-1}^T s_{k-1})}{(s_{k-1}^T y_{k-1})}.[10]$$

Délku kroku α_k, můžeme přepsat také na

$$\alpha_k = g_{k-1}^T g_{k-1} / g_{k-1}^T A g_{k-1},$$

ze které je již patrná návaznost na metodu SD, zatímco ta pracuje s gradientem z daného kroku, metoda BB používá gradientu z kroku předchozího.

Je dána SPD matice $A \in \mathbb{R}^{n\times n}$*, vektor* $b \in \mathbb{R}^n$*, dostatečně malou přesnost* $\epsilon \in \mathbb{R} \wedge \epsilon > 0$ *a* $x_0 \in \mathbb{R}^n$*.*

input: A, b, x_0, ϵ

$g_0 = Ax_0 - b, k = 0$

while $||g_k||/||g_{k_0}|| \leq \epsilon$ **do**

if *k=0* **then**

$\alpha_k = -g_k^T g_k / g_k^T A g_k$

else

$\alpha_k = \frac{(s_{k-1}^T s_{k-1})}{(s_{k-1}^T y_{k-1})}$

end

$x_{k+1} = x_k - \alpha_k g_k$

$s_k = x_{k+1} - x_k$

$g_{k+1} = Ax_{k+1} - b$

$y_k = g_{k+1} - g_k$

$k = k + 1$

end

$\bar{x} = x_k$

Algoritmus 3: Metoda Barzilai-Borwein

Konvergenci této metody pro dobře podmíněnou matici A, kde $\lambda_{\min}$aλmax budou její extrémní vlastní čísla splňující podmínku

$$\lambda_{\max} \leq 2 * \lambda_{\min},$$

je možné vyjádřit pomocí chyby e_{k+1} jako

$$||e_{k+1}|| \leq c||e_k||,$$

kde c bude konvergenční faktor, který můžeme vyjádřit jako

$$c = \frac{\lambda_{\max} - \lambda_{\min}}{\lambda_{min}} \leq 1.[7]$$

3.3 Sdružené gradienty

Oproti již uvedeným algoritmům metoda sdružených gradientů není založena pouze na lineárním modelu, ale využívá tzv. sdružených směrů, díky kterým se její konvergence zrychluje v každém kroce. Definujme si tedy nejdříve sdružené směry a Gramův-Schmidtův ortogonalizační proces, čehož je třeba pro pochopení samotné metody.

Volba sdružených směrů

Množinu nenulových nezávislých n-vektorů $\{p_1, \ldots, p_n\}$ takových, že

$$(p_i, p_j)_A = (p_i)^T A p_j = 0 \qquad \text{pro} \qquad i \neq j.$$

kde A je SPD matice, si definujeme jako množinou sdružených (A-ortogonálních) vektorů, pokud budou jednotlivé vektory na sobě nezávislé a pro libovolné $x \in \mathbb{R}^n$ bude platit, že

$$x = \gamma_1 p_1 + \cdots + \gamma_n p_n.$$

Využitím těchto vektorů dostaneme po substituci do funkce f 2.0.1

$$f(x) = (\tfrac{1}{2}\gamma_1^2 (p_1)^T A p_1 - \gamma_1 b^T p_1) + \cdots + (\tfrac{1}{2}\gamma_n^2 (p_n)^T A p_n - \gamma_n b^T p_n) = f(\gamma_1 p_1) + \cdots + f(\gamma_n p_n),$$

odkud pro řešení $\bar{x}$ platí, že

$$f(\bar{x}) = \min_{x \in \mathbb{R}^n} f(x) = \min_{\gamma_1 \in \mathbb{R}} f(\gamma_1 p_1) + \cdots + \min_{\gamma_n \in \mathbb{R}} f(\gamma_n p_n).$$

Původní úlohu 2.1.1 nyní můžeme rozložit na n jednorozměrných úloh, jelikož

$$\tfrac{\mathrm{d} f(\gamma p_i)}{\mathrm{d}\gamma}|_{\gamma_i} = \gamma_i (p_i)^T A p_i - b^T p_i = 0,$$

řešení $\bar{x}$ úlohy 2.1.1 můžeme přespat na

$$\bar{x} = \gamma_1 p_1 + \cdots + \gamma_n p_n, \qquad \gamma_i = b^T p_i / (p_i)^T A p_i, \qquad i = 1, \ldots, n.$$

Opět však budeme narážet na problém nalezení přesného řešení v případě velké dimenze úlohy. Proto se pokusíme nalézt aproximaci $\tilde{x}$ k řešení $\bar{x}$ založené na volbě x_0 a několika vektorů $p_1, \ldots, p_n$, $k \ll n$. Přirozenou volbou je volba aproximace je minimum x_k funkce $f \in S_k = x_0 + \mathrm{Span}\{p_1, \ldots, p_k\}$, kde *Span* značí lineární obal.

Poznámka 4 Prvek lineárního obalu $x \in S_k$ je možné vyjádřit pomocí

$$x = x_0 + \gamma_1 p_1 + \cdots + \gamma_k p_k.$$

Po dosazení sdružených směrů do funkce f dostaneme

$$f(x) = f(x_0) + (\tfrac{1}{2}\gamma_1^2 (p_1)^T A p_1 - \gamma_1 (A x_0 - b)^T p_1) + \cdots + (\tfrac{1}{2}\gamma_k^2 (p_k)^T A p_k - \gamma_k (A x_0 - b)^T p_k),$$

přičemž z rovnic $g_0 = g(x_0) = \nabla f(x_0) = A x_0 - b$ a

$$f_0(x) = \tfrac{1}{2} x^T A x + x^T g_0$$

dostaneme, že

$$f(x) = f(x_0) + f_0(\gamma_1 p_1) + \cdots + f_0(\gamma_k p_k)$$

a

$$f(x_k) = \min_{x \in S_k} f(x) = f(x_0) + \min_{\gamma_1 \in \mathbb{R}} f_0(\gamma_1 p_1) + \cdots + \min_{\gamma_k \in \mathbb{R}} f_0(\gamma_k p_k).$$

Úlohu se nám tak povedlo zjednodušit a její řešení pomocí aproximace x_k nyní můžeme vyjádřit jako

$$x_k = x_0 + \gamma_1 p_i + \cdots + \gamma p_k, \qquad \gamma_i = -(g_0)^T p_i / (p_i)^T A p_i, \qquad i = 1, \ldots, k, \tag{3.3.1}$$

jelikož

$$(\tfrac{\mathrm{d} f(\gamma p_i)}{\mathrm{d}\gamma})|_{\gamma_i} = \gamma_i (p_i)^T A p_i - (g_0)^T p_i = 0.$$

Z rovnice 3.3.1 tak pro $k \geq 1$

$$f(x_k) = \min_{x \in S_k} f(x) = f(x_{k-1}) + \min_{\gamma \in \mathbb{R}} f_0(\gamma p_k)$$

dostaneme postupné aproximace x_k, které budou začínat od počátečního odhadu x_0 a bude pro ně platit předpis

$$x_k = x_{k-1} - \alpha p_k, \qquad \alpha_k = (g_0)^T p_i / (p_i)^T A p_i.$$

Jelikož $f(x_{k-1} + \gamma p_k)$ nabývá minima, když $\gamma = -\alpha_k$, a metoda zaručuje, že postupné iterace x_k minimalizují f na podprostoru S_k.

Dosud jsme si popsali spíše užití sdružených směrů než to, jak konkrétně by měly být zvoleny. Pojďme se tedy zaměřit na generování těchto směrů, přičemž klíčové bude využití Gramova-Schmidtova procesu. Budeme tedy předpokládat, že $p_1, \ldots, p_k, 1 \leq k \leq n$ jsou nenulové sdružené směry, do jejichž lineárního obalu nebude patřit vektor $h_k, h_k \notin \mathrm{Span}\{p_1, \ldots, p_k\}$, a pokusíme se nalézt nový prvek p_{k+1} jako

$$p_{k+1} = h_k + \beta_{k1} p_1 + \cdots + \beta_{kk} p_k. \tag{3.3.2}$$

Samozřejmě musí být p_{k+1} sdružené k vektorům $p_1, \ldots, p_k$, tudíž bude platit

$$\begin{aligned} 0 = (p_i)^T A p_{k+1} &= (p_i)^T A h_k + \beta_{k1} (p_i)^T A p_1 + \cdots + \beta_{kk} (p_i)^T A p_k \\ &= (p_i)^T A h_k + \beta_{ki} (p_i)^T A p_i, \qquad i = 1, \ldots, k. \end{aligned}$$

Z čehož můžeme odvodit

$$\beta_{ki} = -\tfrac{(p_i)^T A h_k}{(p_i)^T A p_i}, \qquad i = 1, \ldots, k.$$

Pak tedy

$$\mathrm{Span}\{p_1, \ldots, p_k, p_{k+1}\} = \mathrm{Span}\{p_1, \ldots, p_k, h_k\}.$$

Tento postup je možné zopakovat pro libovolné nezávislé vektory $h_0, \ldots, h_{k-1}$, kde počátkem bude $p_1 = h_0$. Vytvoříme tím množinu vzájemně A-sdružených směrů k $p_1, \ldots, p_k$

$$\text{Span}\{h_0, \ldots, h_{i-1}\} = \text{Span}\{p_1, \ldots, p_k\}, \qquad i = 1, \ldots, k.$$

Jediným úskalím tohoto postupu je jeho implementace, při které by docházelo k náročným násobením. Proto se ještě využívá sdružené báze Krylova prostoru

$$K_k = K_k(A, g_0) = \text{Span}\{g_0, Ag_0, \ldots, A_{k-1}g_0\}, \qquad k = 1, \ldots, n,$$

kde $g_0 = Ax_0 - b$ pro vhodné x_0 a $K_0 = \{o\}$. Nyní budeme předpokládat, že $p_1, \ldots, p_i$ tvoří sdruženou bázi prostoru $K_i, i = 1, \ldots, k$, a že gradient $g_k = \nabla f(x_k)$ je ortogonální Krylovu prostoru K_k, tedy

$$(g_k)^T x = 0 \qquad \text{pro libovolné } x \in K_k.$$

Poznámka 5 Pokud tedy $g_k \neq o$,pak

$$g_k \notin K_k.$$

Můžeme si všimnout, že bude platit také pro libovolné $x \in K_k$ pro $k \geq 1$, že

$$Ax \in K_k,$$

nebo obecně $AK_{k-1} \subset K_k$. Jelikož také $p_i \in K_i \subset K_{k-1}$, $i = 1, \ldots, k-1$, dostaneme, že

$$(Ap_i)^T g_k = (p_i)^T A g_k = 0, \qquad i = 1, \ldots, k-1.$$

Odkud plyne

$$\beta_{ki} = -\frac{(p_i)^T A g_k}{(p_i)^T A p_i} = 0, \qquad i = 1, \ldots, k-1.$$

Pro shrnutí, množina sdružených vektorů $p_1, \ldots, p_k$ takových, že

$$\text{Span}\{p_1, \ldots, p_i\} = K_i, \qquad i = 1, \ldots, k,$$

po dosazení do přepisu 3.3.2 s parametrem $h_k = g_k$, který sdruženou bázi, vede k

$$p_{k+1} = g - k + \beta_k p_k \tag{3.3.3}$$

a

$$\beta_k = \beta_{kk} = -\frac{(p_k)^T A g_k}{(p_k)^T A p_k}. \tag{3.3.4}$$

Konečně díky ortogonalitě g_k na $\text{Span}\{p_1, \ldots, p_k\}$ a 3.3.3 dostaneme, že

$$||p_{k+1}|| \geq ||g_k||.$$

Poznámka 6 Pokud tedy $g_{k-1} \neq o$, pak ani $p_k \neq o$, což potvrzuje dobře utvořený předpis pro β_k.

3.3.1 Metoda sdružených gradientů

Metoda sdružených gradientů je se zakládá na výsledcích dvou pozorování a to, že pomocí sdružených směrů jsme schopni konvexní úlohu QP rozložit na řešení posloupnosti jednodimenzionálních úloh a že jsme schopni tyto směry generovat velmi efektivně.

Vstupními parametry metody jsou počáteční odhad x_0, gradient$g_0 = Ax_0 - b$ a vektor $p_1 = g_0$. Pro $k \geq 1$ při daných x_{k-1} a g_{k-1} se přesvědčíme, zda x_{k-1} není řešením, pokud ne, algoritmus vygeneruje

$$x_k = x_{k-1} - \alpha_k p_k \text{ spolu s } \alpha_k = (g_k)^T p_k / (p_k)^T A p_k$$

a následně

$$\begin{aligned} g_k &= Ax_k - b = A(x_{k-1} - \alpha_k p_k) - b = (Ax_{k-1} - b) - \alpha_k A p_k \\ &= g_{k-1} - \alpha A p_k. \end{aligned}$$

Nový sdružený směr p_k pak vzejde z rovnice 3.3.3 a 3.3.4.

Je dána SPD matice $A \in \mathbb{R}^{n \times n}$, vektor $b \in \mathbb{R}^n$, dostatečně malou přesnost $\epsilon \in \mathbb{R} \wedge \epsilon > 0$ a $x_0 \in \mathbb{R}^n$.

input: A, b, x_0, ϵ

$g_0 = Ax_0 - b, z_0 = M^{-1}g, p_1 = z_0, k = 1$

while $||g_k||/||g_{k_0}|| \leq \epsilon$ **do**

$\alpha_k = ||g_{k-1}||^2/(p_k)^T A p_k$

$x_k = x_{k-1} - \alpha_k p_k$

$g_k = g_{k-1} - \alpha_k A p_k$

$z_k = M^{-1} g_k$

$\beta_k = ||g_k||^2/||g_{k-1}||^2 = -(Ap_k)^T g_k/((p_k)^T A p_k)$

$p_{k+1} = g_k + \beta_k p_k$

$k = k + 1$

end

$\bar{x} = x_k$

Algoritmus 4: Metoda sdružených gradientů

Odhad rychlosti konvergence při volbě $x_0 \in \mathbb{R}^n$ u této metody je možné vyjádřit jako

$$||e_k||_A \leq 2(\frac{\sqrt{\kappa(A)}-1}{\sqrt{\kappa(A)}+1})_k ||e_0||_A,$$

kde $e_k = x_k - \bar{x}$ je chyba v dané iteraci a P_k je množinou polynomů k-tých stupňů (poz. $p(0) = 1$).[1]

4 Předpodmínění gradientních metod

Při analýze konvergence jednotlivých gradientních metod je patrné, že se konvergence odvíjí od rozložení vlastních čísel matice A, která je hessiánem funkce f. V případě, že vlastní čísla matice A jsou rozložena na okolí jednoho bodu, případně číslo podmíněnosti $\kappa(A)$ se blíží hodnotě jedna, je konvergence obecně rychlejší. V této kapitole si ukážeme, že při užití vhodné matice M takové, že $M^{-1}x$ je možné snadno vyčíslit pro libovolné x a M aproximuje A tak, že $M^{-1}A$ je blízké matici identity I, jsme schopni úlohu minimalizace ještě zjednodušit. Běžně je cílem takového předpodmínění zmenšení čísla podmíněnosti úlohy.

Matice předpodmínění

Mějme matici M ve formě

$$M = \tilde{L}\tilde{L}^T,$$

přičemž $M^{-1}A$ je blízké $\tilde{L}^{-1}A\tilde{L}^{-T}$ a výsledná matice pak je blízká matici identity. Po dosazení do předpisu kvadratické funkce f 2.0.1 dostaneme

$$f(x) = \tfrac{1}{2}(\tilde{L}^T x)^T(\tilde{L}^{-1}A\tilde{L}^{-T})(\tilde{L}^T x) - (\tilde{L}^{-1}b)^T(\tilde{L}^T x)$$

a naši původní úlohu 2.1.1 si tak převedeme na předpodmíněnou úlohu

$$\min_{y\in\mathbb{R}^n} \bar{f}(y), \tag{4.0.1}$$

pro kterou jsme užili substituci $y = \tilde{L}^T x$, její kvadratický předpis tak má podobu

$$\bar{f}(y) = \tfrac{1}{2}y^T(\tilde{L}^{-1}A\tilde{L}^{-T})y - (\tilde{L}^{-1}b)^T y.$$

Řešení původní úlohy tak z té nové (4.0.1) dostaneme při známém řešení $\bar{y}$ jako

$$\bar{x} = \tilde{L}^{-T}\bar{y}.$$

Předpodmínění SSOR

V případě předpodmínění SSOR je matice M vyjádřena pomocí

$$M = (D + L)D^{-1}(D + L)^T,$$

což vychází z rozkladu matice A na diagonální, horní a dolní trojúhelníkovou matici, tedy

$$A = D + L + L^T.$$

Toto předpodmínění vychází z metody SOR (succesive over-relaxation), která je adaptací Gauss-Seidelovy metody k řešení lineární soustavy rovnic, na kterých zaznamenává rychlejší konvergenci než původní verze. Pro SSOR se využívá symetrie rozkládané matice a při jeho vhodné implementaci není rozklad výpočetně náročný.

4.1 Předpodmíněná metoda největšího spádu

Pokud bychom použili metodu SD k řešení úlohy 4.0.1, za předpokladu známého y_0 bychom nejprve definovali

$$y_0 = \tilde{L}^T x_0, \qquad \bar{g}_0 = \tilde{L}^{-1} A \tilde{L}^{-T} y_0 - \tilde{L}^{-1} b = \tilde{L}^{-1} g_0$$

dále pak

$$\begin{aligned} \bar{\alpha}_k &= ||\bar{g}_{k-1}||^2 / (\bar{g}_k^T \tilde{L}^{-1} A \tilde{L}^{-T} \bar{g}_k), \\ y_k &= y_{k-1} - \bar{\alpha}_k \bar{g}_k, \\ \bar{g}_k &= \bar{g}_{k-1} - \bar{\alpha}_k \tilde{L}^{-1} A \tilde{L}^{-T} \bar{g}_k, \end{aligned}$$

Z následující substituce

$$y_k = \tilde{L}^T x_k, \qquad \bar{g}_k = \tilde{L}^{-1} \bar{g}_k$$

a definice

$$z_k = \tilde{L} \tilde{L}^T g_k = M^{-1} g_k$$

dostaneme předpodmíněnou metodu největšího spádu (PSD), někdy také nazývanou jako gradientní metodu s dynamickým parametrem.[9]

Je dána SPD matice $A \in \mathbb{R}^{n \times n}$, její aproximace $M \in \mathbb{R}^{n \times n}$, vektor $b \in \mathbb{R}^n$, dostatečně malou přesnost $\epsilon \in \mathbb{R} \wedge \epsilon > 0$ a $x_0 \in \mathbb{R}^n$.

input: A, b, x_0, ϵ

$g_0 = Ax_0 - b, z_0 = M^{-1} g_0, k = 0$

while $||g_k|| / ||g_{k_0}|| \leq \epsilon$ **do**

$\alpha_k = g_k^T z_k / z_k^T A z_k$

$x_{k+1} = x_k - \alpha_k z_k$

$g_{k+1} = g_k - \alpha_k A z_k$

$z_{k+1} = M^{-1} g_{k+1}$

$k = k + 1$

end

$\bar{x} = x_k$

Algoritmus 5: Předpodmíněná metoda největšího spádu (PSD)

4.2 Předpodmíněná metoda Barzilai-Borwein

V případě užití metody BB k řešení úlohy 4.0.1 je předefinování proměnných obdobné jako v případě metody SD, výjimkou je samozřejmě délka kroku, která bude opět počítat se směry z kroku předchozího

$$\alpha_k = g_{k-1}^T z_{k-1} / z_{k-1}^T A z_{k-1},$$

kde

$$z_k = \tilde{L}\tilde{L}^T g_k = M^{-1} g_k$$

dostaneme předpodmíněnou metodu Barzilai-Borwein (PBB). [7]

Je dána SPD matice $A \in \mathbb{R}^{n\times n}$, její aproximace $M \in \mathbb{R}^{n\times n}$, vektor $b \in \mathbb{R}^n$, dostatečně malou přesnost $\epsilon \in \mathbb{R} \wedge \epsilon > 0$ a $x_0 \in \mathbb{R}^n$.

input: A, b, x_0, ϵ

$g_0 = Ax_0 - b, z_0 = M^{-1} g_0, k = 0$

while $||g_k||/||g_{k_0}|| \leq \epsilon$ **do**

$\alpha_k = g_{k-1}^T z_{k-1} / z_{k-1}^T A z_{k-1}$

$x_{k+1} = x_k - \alpha_k z_k$

$g_{k+1} = g_k - \alpha_k A z_k$

$z_{k+1} = M^{-1} g_{k+1}$

$k = k + 1$

end

$\bar{x} = x_k$

Algoritmus 6: Předpodmíněná metoda Barzilai-Borwein(PBB)

4.3 Předpodmínění metody sdružených gradientů

Pokud bychom použili metodu CG k řešení úlohy 4.0.1, za předpokladu známého y_0 bychom nejprve definovali

$$y_0 = \tilde{L}^T x_0, \qquad \bar{g}_0 = \tilde{L}^{-1} A \tilde{L}^{-T} y_0 - \tilde{L}^{-1} b = \tilde{L}^{-1} g_0, \qquad \bar{p} = \bar{g}_0$$

dále pak

$$\begin{aligned}
\bar{\alpha}_k &= ||\bar{g}_{k-1}||^2 / (\bar{p}_k^T \tilde{L}^{-1} A \tilde{L}^{-T} \bar{p}_k), \\
y_k &= y_{k-1} - \bar{\alpha}_k \bar{p}_k, \\
\bar{g}_k &= \bar{g}_{k-1} - \bar{\alpha}_k \tilde{L}^{-1} A \tilde{L}^{-T} \bar{p}_k, \\
\bar{\beta}_k &= ||\bar{g}_k||^2 / ||bar g_{k-1}||^2, \\
\bar{p}_{k+1} &= \bar{g}_k + \bar{\beta}_k \bar{p}_k.
\end{aligned}$$

Z následující substituce

$$y_k = \tilde{L}^T x_k, \qquad \bar{g}_k = \tilde{L}^{-1} \bar{g}_k, \qquad \bar{p}_k = \tilde{L}^T p_k$$

a definice

$$z_k = \tilde{L}\tilde{L}^T g_k = M^{-1} g_k$$

vyplývá předpodmíněná metoda sdružených gradientů (PCG).

Je dána SPD matice $A \in \mathbb{R}^{n \times n}$*, její aproximace* $M \in \mathbb{R}^{n \times n}$*, vektor* $b \in \mathbb{R}^n$*, dostatečně malou přesnost* $\epsilon \in \mathbb{R} \wedge \epsilon > 0$ *a* $x_0 \in \mathbb{R}^n$

input: A, b, x_0, ϵ

$g_0 = Ax_0 - b, z_0 = M^{-1}g_0, p_1 = z_0, k = 1$

while $||g_k||/||g_{k_0}|| \leq \epsilon$ **do**

$\alpha_k = (z_{k-1})^T g_{k-1}/(p_k)^T A p_k$

$x_k = x_{k-1} - \alpha_k p_k$

$g_k = g_{k-1} - \alpha_k A p_k$

$z_k = M^{-1} g_k$

$\beta_k = (z_k)^T g_k/(z_{k-1})^T g_{k-1}$

$p_{k+1} = z_k + \beta_k p_k$

$k = k + 1$

end

$\bar{x} = x_k$

Algoritmus 7: Předpodmíněná metoda sdružených gradientů (PCG)

5 Úlohy pro porovnání gradientích metod

K porovnání metod si vytvoříme několik modelových minimalizačních úloh bez omezení. Cílem bude porovnat pro různé podmínky zadání úloh rychlost konvergence jednotlivých metod, přičemž pro všechny úlohy bude platit, že $A \in \mathbb{R}^{n\times n}$ je SPD matice, požadovaná ukončovací podmínka přesnosti řešení $\epsilon = 10^{-6}$, pro $b \in \mathbb{R}^n$ jakožto vektor pravých stran bude platit

$$b = [1, \ldots, 1]^T$$

a volbu počátečního odhadu $x_0 \in \mathbb{R}^n$

$$x_0 = [0, \ldots, 0]^T.$$

5.1 Minimalizační úlohy QP bez omezení a porovnání gradientních metod bez předpodmínění

U první z úloh se zaměříme na průběh proměnné x_k, který si pro názornost vykreslíme na vrstevnicích funkce, z čehož plyne, že dimenze matice A bude nutně $n = 2$, pro jinou dimenzi bychom toho nebyli schopni.

Příklad 1
Nalezněme řešení soustavy $Ax = b$ při zadané matici o $n = 2$

$$A = \begin{pmatrix} 19 & 15 \\ 15 & 27 \end{pmatrix}$$

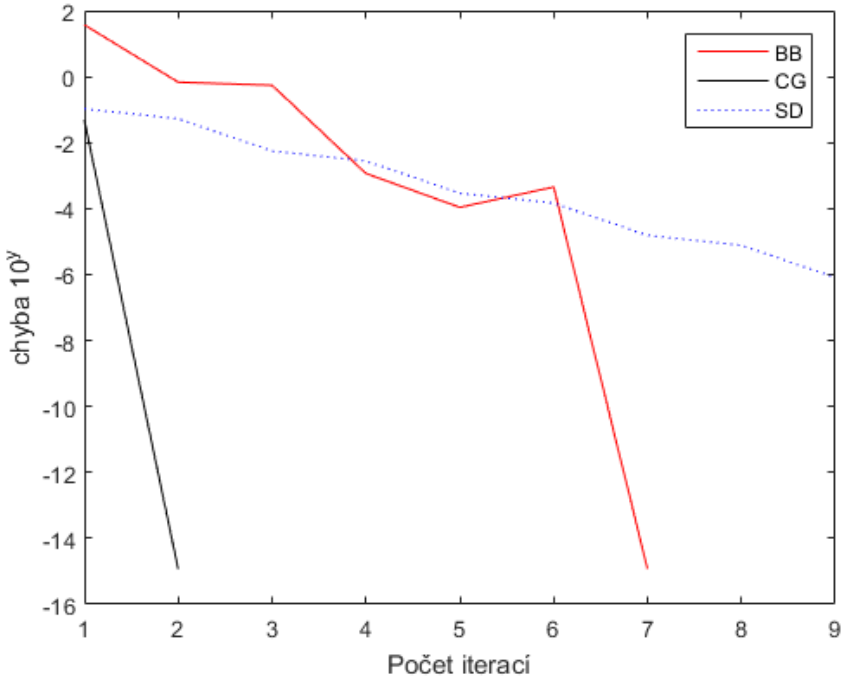
Po provedení výpočtu dosáhly řešení v tomto případě při stanovené přesnosti všechny metody, jak je vidět na prvním grafu (1). Číslo podmíněnosti dané matice je $\kappa(A) \approx 5,1532$, tedy matice je dobře podmíněná a při dimenzi $n = 2$ také malá.

Na zbývajících grafech je vidět průběh iterací x_k jednotlivých metod, nejrychleji k dostatečně přesnému řešení konvergovala metoda CG, zatímco nejpomaleji ho dosáhla metoda SD, což není překvapením, odpovídá to jejich odhadu rychlosti konvergence.

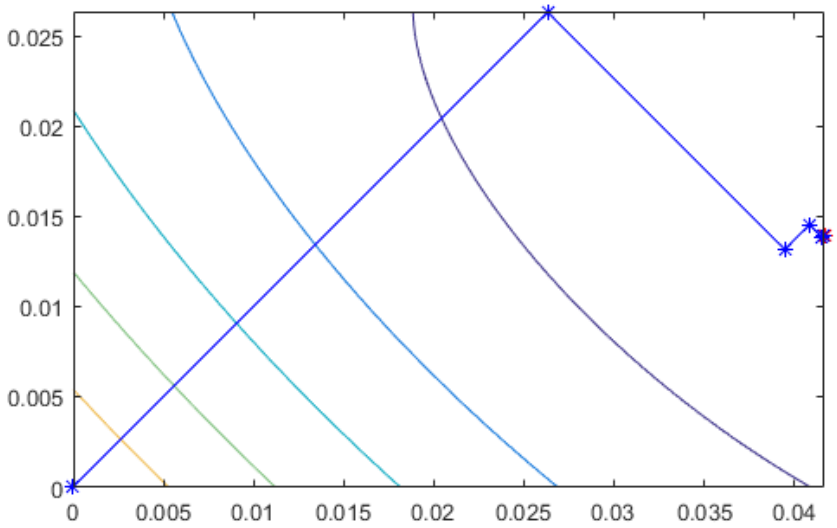
Všimněme si ještě, že metoda CG dosáhla po druhé iteraci přesnosti $\approx 10^{-16}$, která odpovídá počítačové přesnosti výpočtů, jinak řečeno metoda CG potřebovala 2 iterace pro dosažení přesného řešení.

Tabulka 5.1.1: Počet iterací gradientních metod pro první příklad

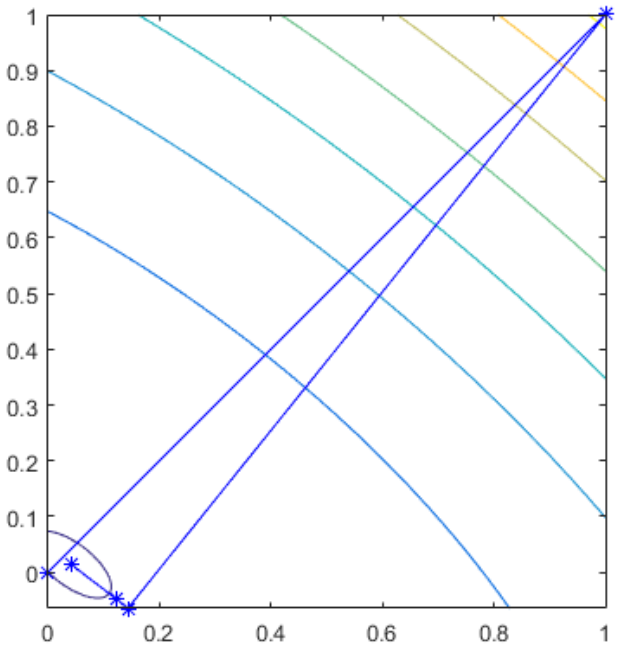
	SD	BB	CG
k	9	7	2



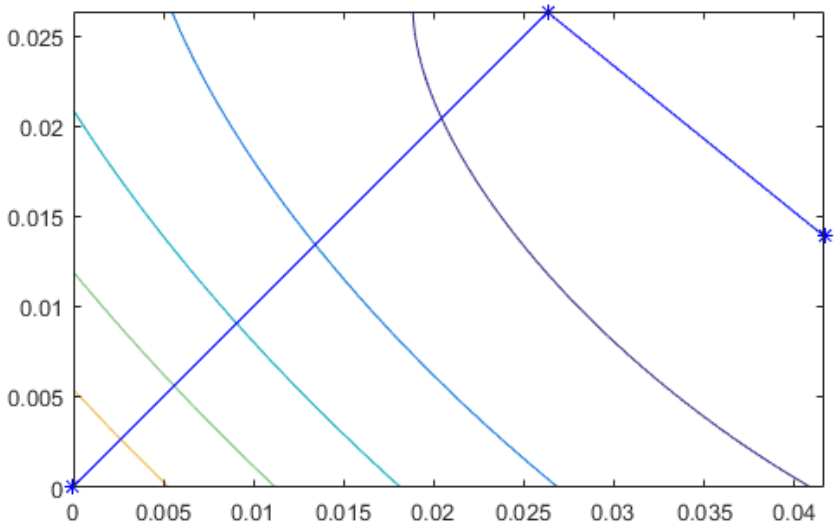
Obrázek 1: Porovnání konvergence gradientních metod pro první příklad



Obrázek 2: Průběh iterací proměnné x_k metody SD pro první příklad



Obrázek 3: Průběh iterací proměnné x_k metody BB pro první příklad



Obrázek 4: Průběh iterací proměnné x_k metody CG pro první příklad

■

U druhé modelové úlohy se zaměříme pouze na konvergenci metod, sledovat budeme průběh relativní velikosti gradientu, tj. $||g_k||/||g_0||$. Zvolíme si Laplaceovu matici, která nachází využití v teorii grafů nebo pro metodu sítí (konečných diferencí). Naše konkrétní matice bude pro metodu sítí reprezentovat strunu, kde každý z prvků má vazbu na prvek sousední. Vektor b bude v tomto případě chápán jako vyjádření síly působící na strunu. Řešení úlohy pak má význam průhybu struny s nulovými Dirichletovými okrajovými podmínkami.

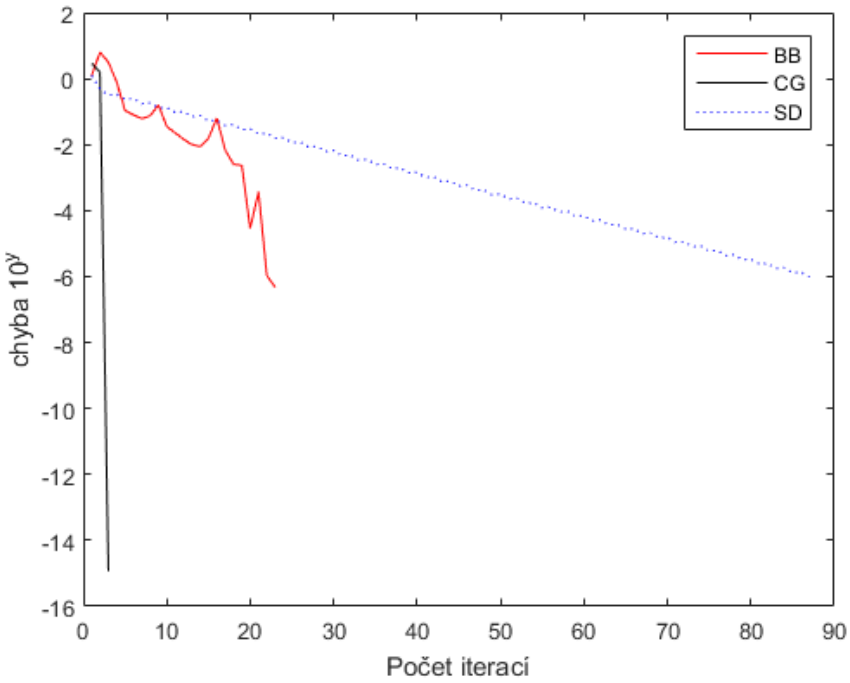
Příklad 2
Nalezněme řešení soustavy $Ax = b$ při zadané matici o $n = 5$

$$A = \begin{pmatrix} 2 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & 2 \end{pmatrix}$$

Číslo podmíněnosti pro tuto matici je $\kappa(A) \approx 13,9282$, jedná se tedy o stále dobře podmíněnou malou matici, proto také vlastnosti konvergence jednotlivých metod, jak je vidět na grafu (5), se nijak výrazně oproti prvnímu příkladu nezměnily, rychlosti konvergencí zůstaly ve stejném pořadí a všechny metody našly řešení dle daných podmínek.

Tabulka 5.1.2: Počet iterací gradientních metod pro druhý příklad

	SD	BB	CG
k	87	23	3



Obrázek 5: Porovnání konvergence gradientních metod pro druhý příklad

■

Pro třetí příklad zůstaneme u Laplaceovy matice, zachováme její definici, avšak tentokrát zvolíme větší dimenzi n. O vlastnostech vlastních čísel této Laplaceovy matice je známo, že jsou rozložena mezi hodnotami 0 a 4. Při zvyšující se dimenzi matici tak dochází k tomu, že $\lim_{n\to\infty}\lambda_{\min} = 0$, z čehož plyne pro číslo podmíněnosti matice $\kappa(A) = \lambda_{\max}/\lambda_{\min}$, že s rostoucí dimenzí matice roste i je číslo podmíněnosti a matice je tak hůř podmíněná.

Příklad 3

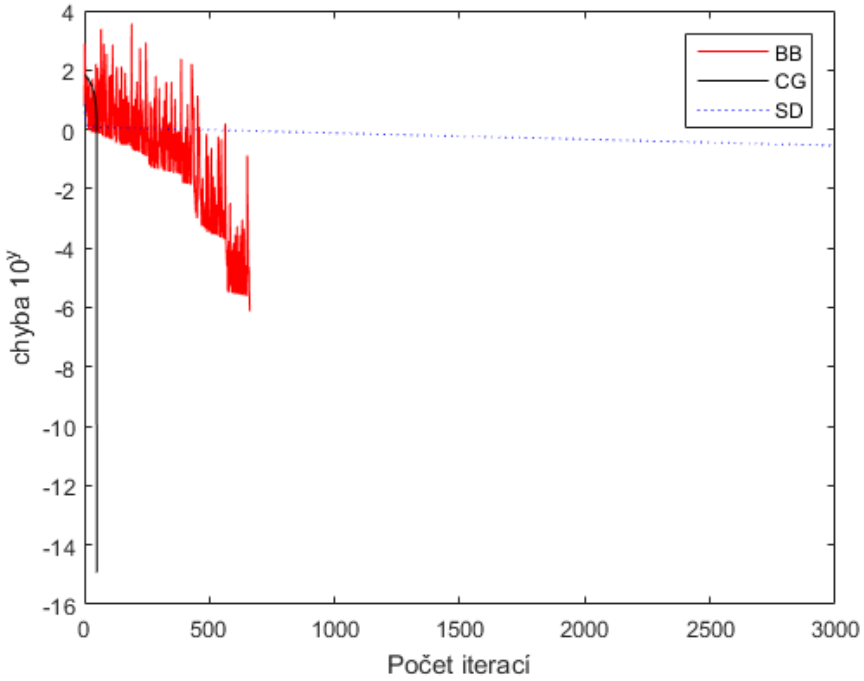
Nalezněme řešení soustavy $Ax = b$ při zadané matici o $n = 100$

$$A = \begin{pmatrix} \ddots & \ddots & 0 & 0 & 0 \\ \ddots & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & \ddots \\ 0 & 0 & 0 & \ddots & \ddots \end{pmatrix}$$

Jak je vidět z grafu průběhu konvergence (6), u této metody byla nastavena dodatečná ukončovací podmínka a to na počet provedených iterací, byla nastavena na pro $k = 3000$. Metoda SD, která daného počtu iterací dosáhla tedy nedokonvergovala k dostatečně přesnému řešení. To je způsobeno právě špatnou podmíněností matice A, jejíž číslo podmíněnosti je tentokrát $\kappa(A) \approx 4133,6$.

Tabulka 5.1.3: Počet iterací gradientních metod pro třetí příklad

	SD	BB	CG
k	3000	661	50



Obrázek 6: Porovnání konvergence gradientních metod pro třetí příklad

■

5.2 Porovnání gradientních metod bez a s předpodmíněním

V této části budeme porovnávat vždy mezi sebou danou metodu bez a s předpodmíněním, přičemž budeme používat předpodmínění SSOR, a porovnáme si i na jednom hromadném grafu pro názornost konvergenci všech metod. K tomuto porovnávání budou použity úlohy z předešlé kapitoly bez jakýchkoliv úprav.

Příklad 4

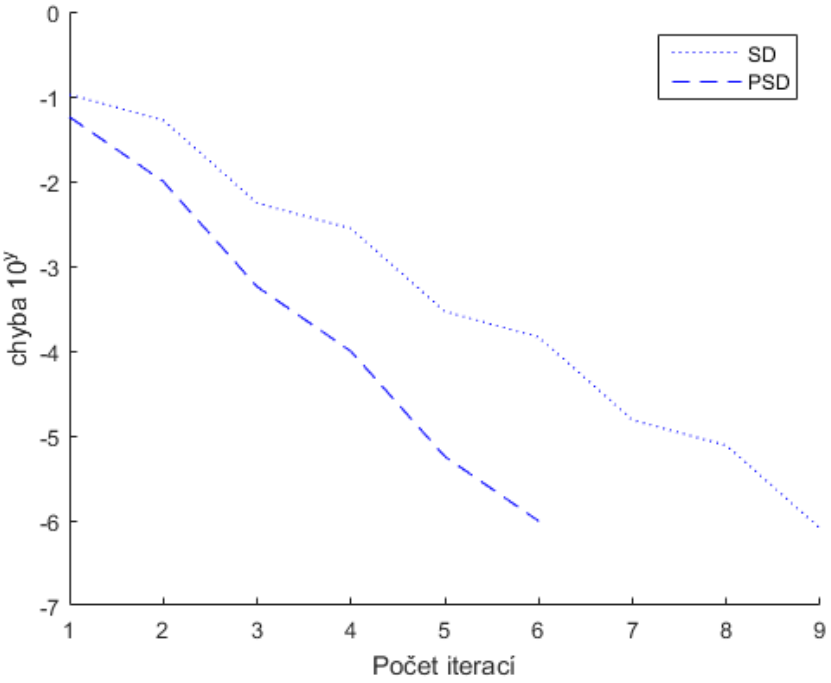
Nalezněme řešení soustavy $Ax = b$ při zadané matici o $n = 2$

$$A = \begin{pmatrix} 19 & 15 \\ 15 & 27 \end{pmatrix}$$

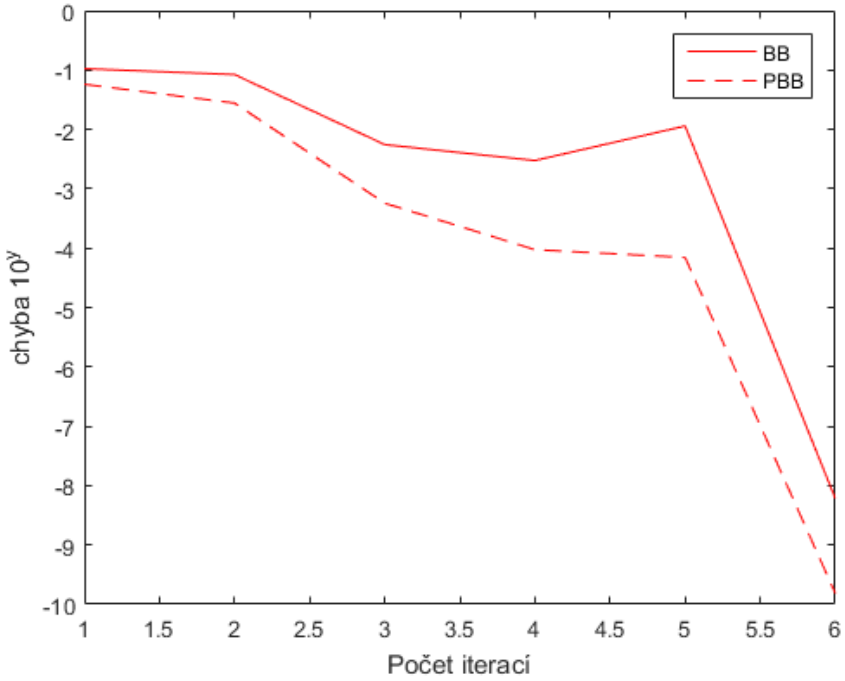
U této úlohy již můžeme zaznamenat zvýšení efektivity gradientní metody díky předpodmínění, metoda PSD dokázala na již tak malé matici urychlit konvergenci o 3 iterace, oproti nepředpodmíněné metodě. U zbylých metod jsme nezaznamenali urychlení konvergence, ale zároveň se její rychlost nezpomalila.

Tabulka 5.2.1: Počet iterací gradientních metod pro čtvrtý příklad

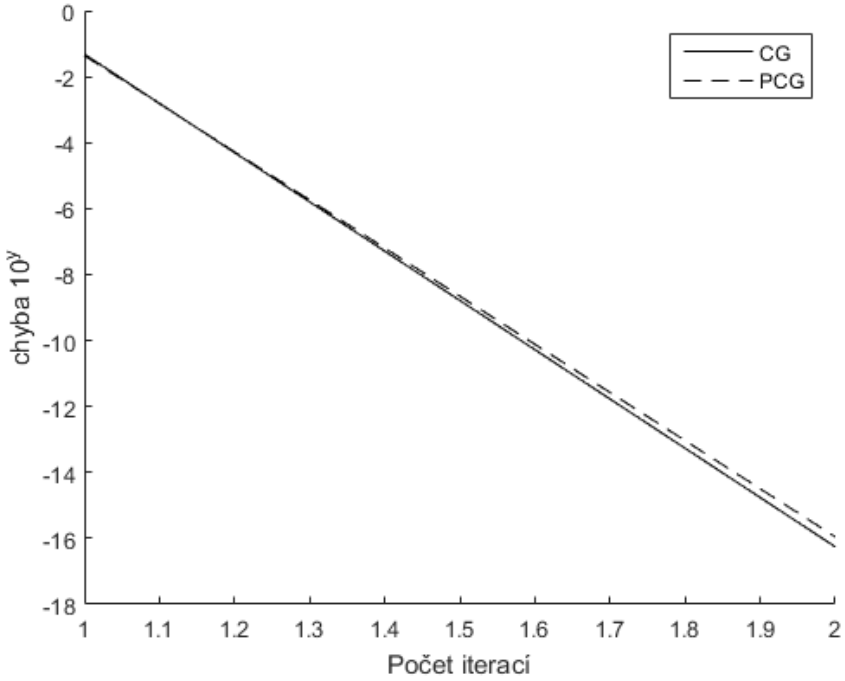
	SD	BB	CG	PSD	PBB	PCG
k	9	6	2	6	6	2



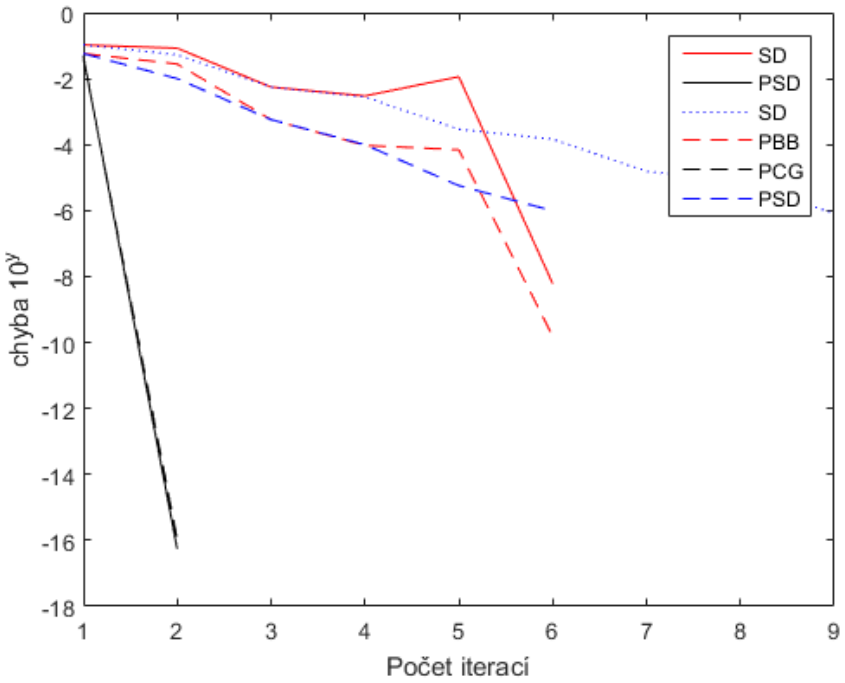
Obrázek 7: Porovnání konvergence metod SD a PSD pro čtvrtý příklad



Obrázek 8: Porovnání konvergence metod BB a PBB pro čtvrtý příklad



Obrázek 9: Porovnání konvergence metod CG a PCG pro čtvrtý příklad



Obrázek 10: Porovnání konvergence vybraných metod pro čtvrtý příklad

■

Příklad 5

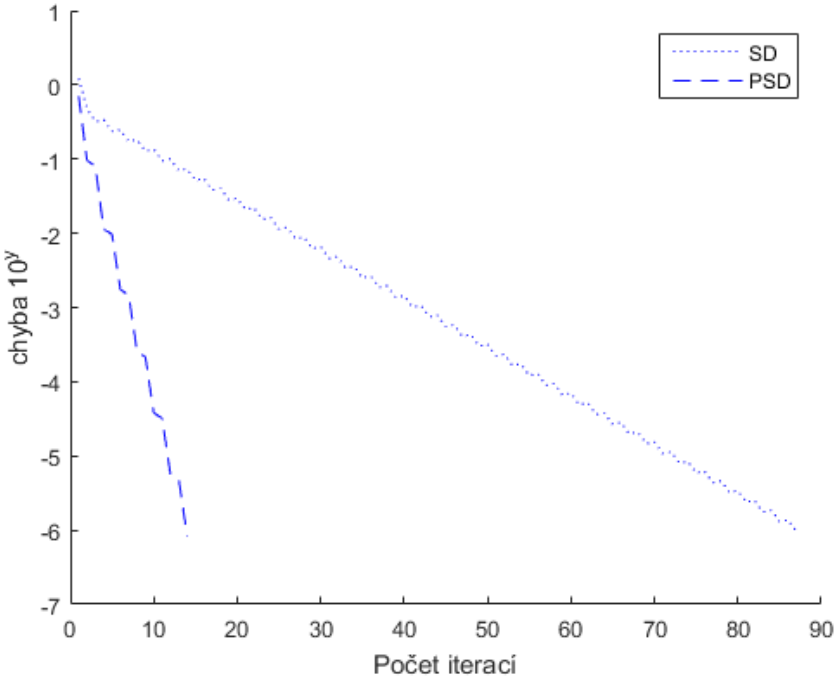
Nalezněme řešení soustavy $Ax = b$ při zadané matici o $n = 5$

$$A = \begin{pmatrix} 2 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & 2 \end{pmatrix}$$

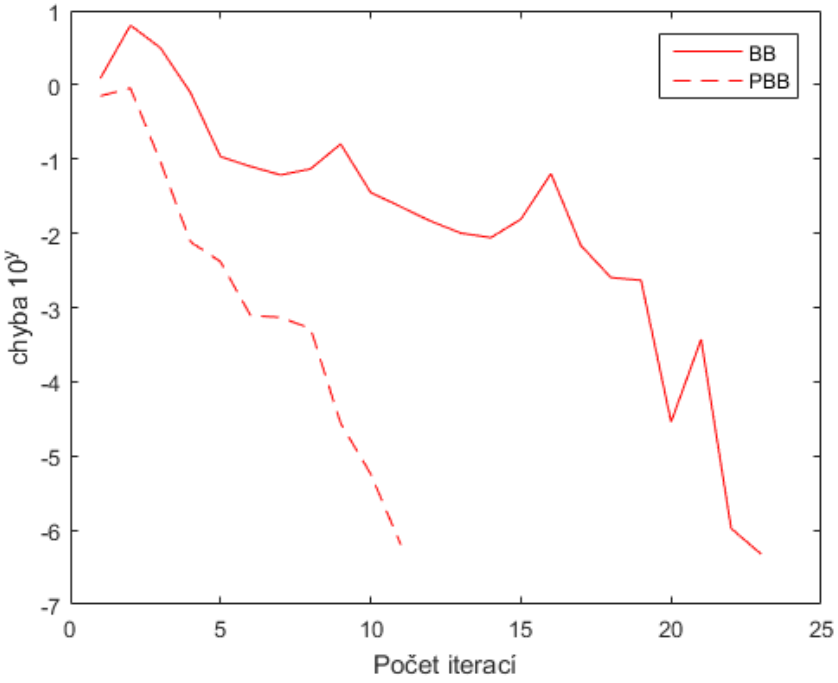
V případě této úlohy mělo předpodmínění rychlejší konvergenci nejen pro metodu PSD, ale také pro metodu PBB oproti jejich nepředpodmíněným verzím. Při této dimenzi Laplaceovy matice však předpodmínění nemělo pozitivní dopad na konvergenci metody PCG, která dokonvergovala o 2 iteraci později než metoda CG, avšak stále byly obě verze metody CG rychlejšími než metody zbylé.

Tabulka 5.2.2: Počet iterací gradientních metod pro pátý příklad

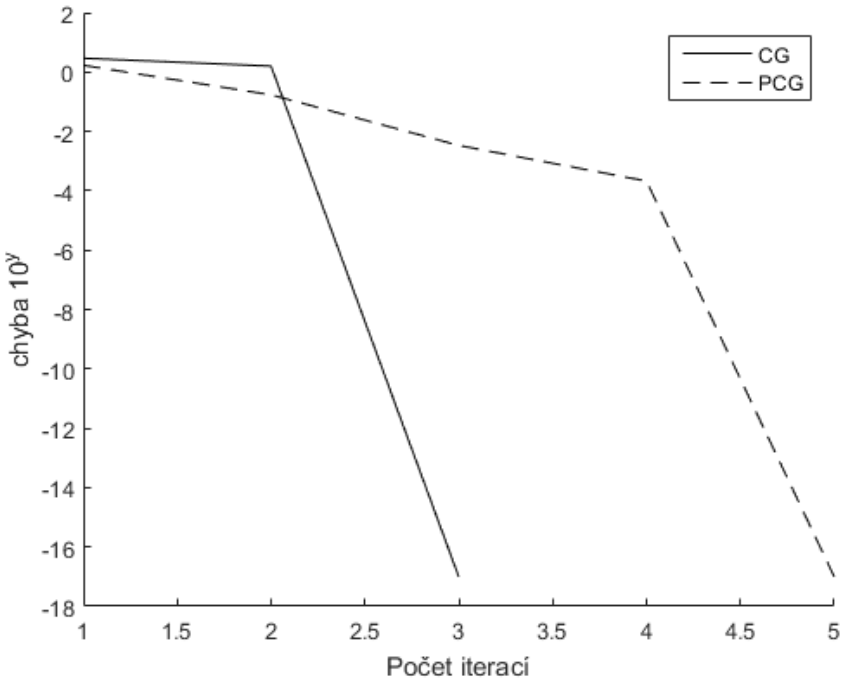
	SD	BB	CG	PSD	PBB	PCG
k	87	23	3	14	11	5



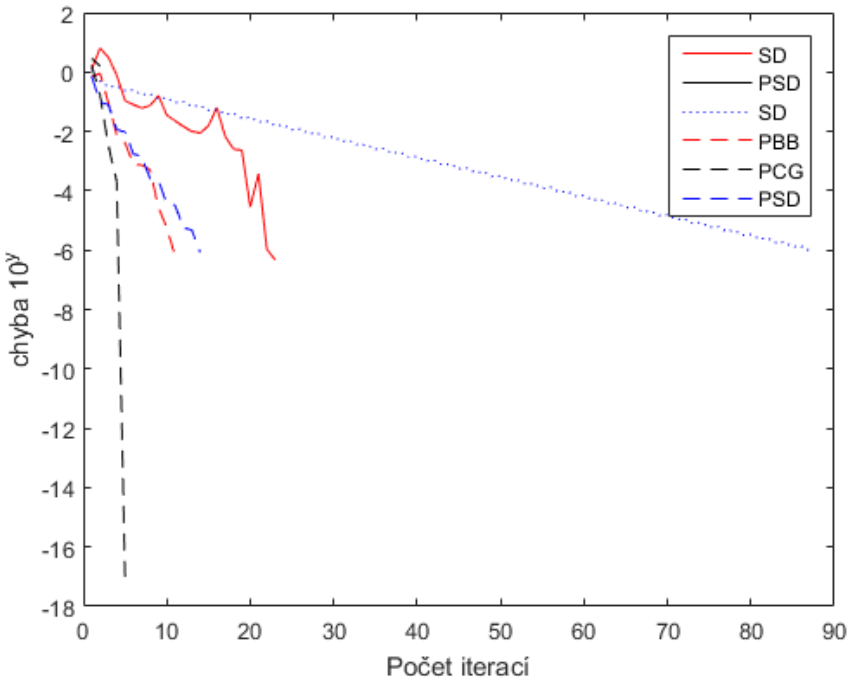
Obrázek 11: Porovnání konvergence metod SD a PSD pro pátý příklad



Obrázek 12: Porovnání konvergence metod BB a PBB pro pátý příklad



Obrázek 13: Porovnání konvergence metod CG a PCG pro pátý příklad



Obrázek 14: Porovnání konvergence vybraných metod pro pátý příklad

■

Příklad 6

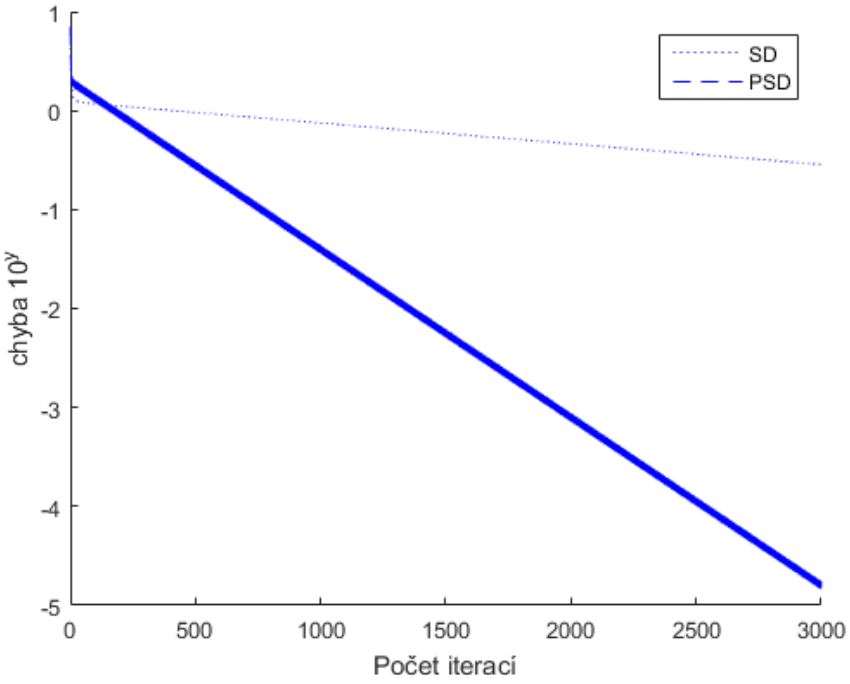
Nalezněme řešení soustavy $Ax = b$ při zadané matici o $n = 100$

$$A = \begin{pmatrix} \ddots & \ddots & 0 & 0 & 0 \\ \ddots & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & \ddots \\ 0 & 0 & 0 & \ddots & \ddots \end{pmatrix}$$

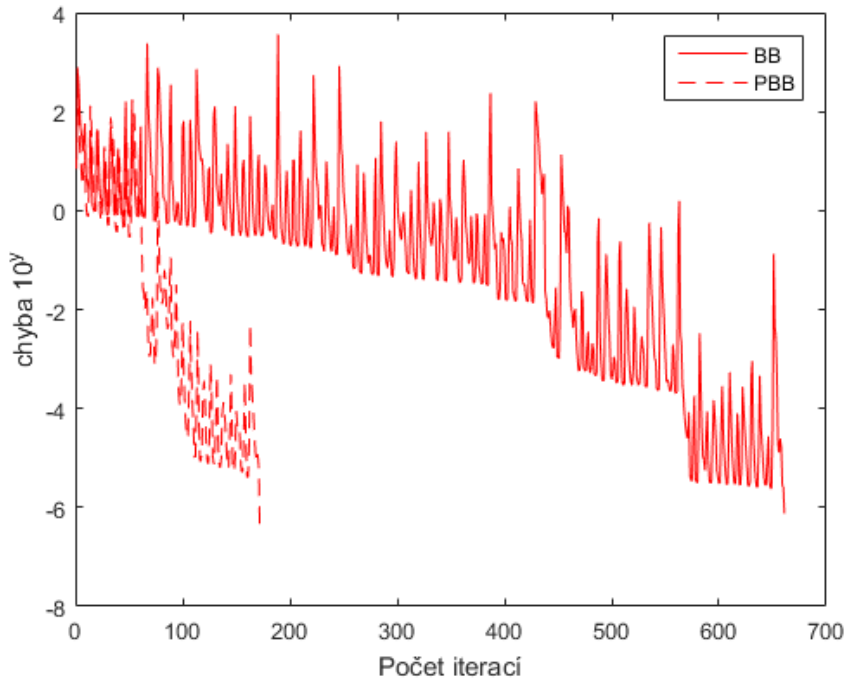
Na poslední úloze je evidentní zrychlení konvergence u všech předpodmíněných metod. Nejviditelnější je pro metodu PSD, která dosáhla přesnosti $\epsilon \approx 10^{-5}$ oproti metodě, která nedosáhla ani přesnosti $\epsilon \approx 10^{-1}$ za omezený počet iterací $k = 3000$. Nejrychleji stejně jako v předešlých úlohách konvergovala metoda sdružených gradientů.

Tabulka 5.2.3: Počet iterací gradientních metod pro šestý příklad

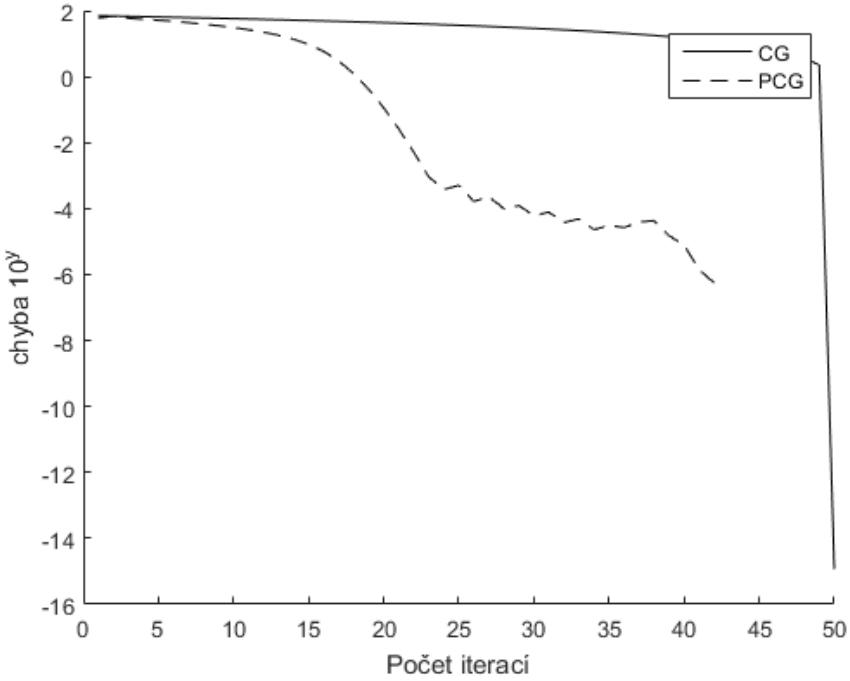
	SD	BB	CG	PSD	PBB	PCG
k	3000	661	50	3000	172	42



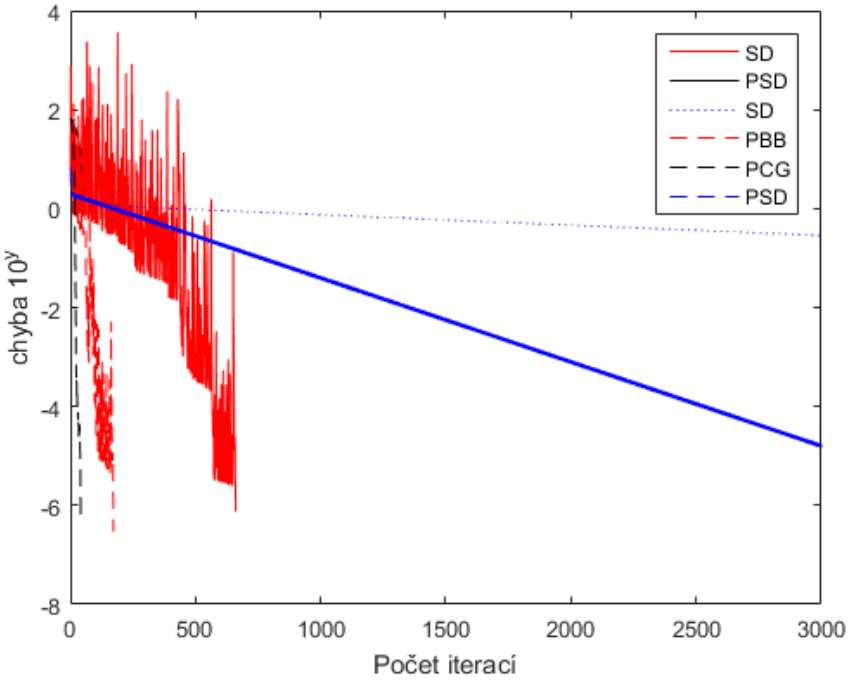
Obrázek 15: Porovnání konvergence metod SD a PSD pro šestý příklad



Obrázek 16: Porovnání konvergence metod BB a PBB pro šestý příklad



Obrázek 17: Porovnání konvergence metod CG a PCG pro šestý příklad



Obrázek 18: Porovnání konvergence vybraných metod pro šestý příklad

■

6 Závěr

Cílem této práce bylo seznámit se s algoritmy kvadratického programování, jejich implementací a aplikací. Proto jsme se nejdříve definovali úlohy kvadratického programování, blíže úlohy bez omezení, včetně podmínek pro nalezení jejich řešení.

Popsali jsme si gradientní metody, nejdříve v obecné rovině, poté jsme navázali konkrétními metodami, kterými jsou metoda největšího spádu, metoda Barzilai-Borwein a metoda sdružených gradientů. U každé z metod jsme si uvedli jejich odvození, jejich předpis algoritmu a také rychlost jejich konvergence k řešení. Jelikož pro vybrané úlohy konvergují gradientní metody rychleji s předpodmíněním, další kapitolu jsme věnovali právě předpodmínění vybraných metod.

V poslední kapitole jsme se dostali k numerickým experimentům, porovnávali jsme vybrané gradientní metody na malých maticích, pro první úlohu se jednalo o dvoudimenzionální, dobře podmíněnou matici, abychom mohli sledovat na vrstevnicích dané funkce vývoj jednotlivých iterací. Pro druhou úlohu jsme si vybrali Laplaceovu matici o dimenzi $n = 5$, stále se jednalo o malou matici a také dobře podmíněnou, avšak tato matice má také další vlastnost a to, že je řídká. Pro poslední úlohu jsme si ponechali Laplaceovu matici, avšak její dimenzi jsme tentokrát zvolili $n = 100$, přičemž se změnila její podmíněnost a jednalo se tak už o špatně podmíněnou matici, což se výrazně projevilo na rychlostech konvergencí.

Numerické experimenty nám potvrdily předpoklady konvergence vybraných metod. Nejrychleji pro všechny úlohy konvergovala metoda sdružených gradientů, následně to byla metoda Barzilai-Borwein a metoda největšího spádu byla ve všech případech nejpomalejší, dokonce pro špatně podmíněnou úlohu nedokonvergovala. Dále se nám povrdilo z experimentů, že předpodmínění může výrazně urychlit konvergenci dané metody. Můžeme z toho vyvodit, že z vybraných algoritmů byla nejefektivnější metoda sdružených gradientů.

Literatura

[1] Dostál, Z., 2009. *Optimal quadratic programming algorithms: with applications to variational inequalities*. Springer.

[2] Kozubek, T., Brzobohatý, T., Hapla, V., Jarošová, M., Markopoulos, A., 2012. *Lineární algebra s Matlabem*. Matematika pro inženýry 21. století (reg. č. CZ.1.07/2.2.00/07.0332). (http://mi21.vsb.cz/modul/linearni-algebra-s-matlabem)

[3] Dostál, Z., Beremlijski, P., 2012. *Metody optimalizace*. Matematika pro inženýry 21. století (reg. č. CZ.1.07/2.2.00/07.0332). (http://mi21.vsb.cz/modul/metody-optimalizace)

[4] Vondrák, V., Pospíšil, L., 2011. *Numerické metody I*. Matematika pro inženýry 21. století (reg. č. CZ.1.07/2.2.00/07.0332). (http://mi21.vsb.cz/modul/numericke-metody-1)

[5] Barzilai,J. , Borwein,J.M., 1988. *Two-point step size gradient methods*. IMA Journal of Numerical Analysis 8.

[6] Dai, Yu-Hong, Fletcher, R., 2003. *Projected Barzilai-Borwein methods for large-scale box-constrained quadratic programming*. Numerical Analysis Report NA/215.

[7] Raydan, Marcos M., 1991. *Convergence properties of the Barzilai and Borwein gradiend method*. Rice University.

[8] Di Serafino, D., Ruggiero,V., Toraldo, G., Zanni,L., 2017. *On the steplength selection in gradient methods for unconstrained optimization*.

[9] Quarteroni, A., Sacco, R., Saleri, F., 2000. *Numerical mathematics*. Springer.

[10] Sun, W., Yuan, Ya-Xiang, 2006. *Optimization Theory and Methods*. Springer.

7 Přílohy

Příloha na CD/DVD:

CG.m
PCG.m
BB.m
PBB.m
SD.m
PSD.m
compare.m
parameters.m
generateL1.m
SSOR.m