Matematicko-fyzikální fakulta

Univerzita Karlova

Praha

Numerická realizace metod vnitřního bodu pro řešení úloh lineárního a kvadratického programování

Věra Koubková

Diplomová práce
Praha 1997

Studijní směr: **Výpočtová matematika – algoritmy**

Vedoucí práce: **Ing. Ladislav Lukšan, DrSc.**

Prohlašuji, že jsem diplomovou práci vypracovala samostatně pouze s použitím uvedené literatury. Souhlasím se zapůjčováním této práce.

Věra Koubková

Mým rodičům za lásku a trpělivost

Ráda bych na tomto místě velmi poděkovala Ing. Ladislavu Lukšanovi, DrSc. za pomoc při vypracování diplomové práce, za zapůjčení literatury a potřebného softwaru a především za čas, který mi věnoval.

Obsah

Úvod

Ačkoli lineární programování coby matematická disciplína vzniklo poměrně nedávno, je v současné době velmi důležitým odvětvím aplikované matematiky. Od prvního zformulování úlohy lineárního programování ve třicátých letech 20. století a od Dantzingova objevu simplexové metody v polovině let čtyřicátých je lineární programování užívané zejména v ekonomických, finančních, ale i technických aplikacích.

Nejvýznamnější událostí od objevu simplexové metody bylo bezesporu opublikování Karmarkarova článku [1] v roce 1984. Zde poprvé byla zmíněna metoda založená na zcela jiném přístupu k problému lineárního programování – projektivní metoda vnitřního bodu. Ohlas, který vzbudila, plynul jednak z jejích teoretických vlastností – metoda dosahovala, na rozdíl od simplexové metody, pouze polynomiální složitosti – a jednak z její praktické využitelnosti pro velké lineární problémy. Karmarkarův článek tak odstartoval další vlnu zájmu o problémy lineárního programování. V současné době existuje nepřeberné množství publikací, které pojednávají o metodách vnitřního bodu založených nejenom na původní Karmarkarově myšlence. Obsahem této práce jsou primárně–duální metody vnitřního bodu, které nejenom že vzbuzují největší zájem teoretiků, ale lze jimi dosáhnout i velmi dobrých praktických výsledků.

Celá práce se zabývá nalezením primárně–duálního řešení úlohy lineárního programování, přičemž se zaměřuje hlavně na jednu třídu primárně–duálních metod; metody path–following.

V první kapitole zformulujeme úlohu LP a definujeme základní pojmy, jako „účelová funkce“ či „optimální řešení“. Poté zavedeme pojem „primární úloha“, „duální úloha“ a dokážeme některá fundamentální tvrzení teorie duality: (silnou) větu o dualitě, Farkasovo lemma, Goldmanovu–Tuckerovu větu a Karushovy–Kuhnovy–Tuckerovy podmínky, které jsou nutnými a postačujícími podmínkami pro nalezení optimálního řešení x^* resp. (y^*, s^*) primární resp. duální úlohy. Na základě těchto podmínek pak definujeme „primárně–duální řešení úlohy LP“ jako vektor (x^*, y^*, s^*) a vytvoříme nelineární funkci F takovou, že $F(x^*, y^*, s^*) = 0$.

Pro řešení soustav $F(x, y, s) = 0$ používáme modifikovanou Newtonovu metodu, jejíž popis je obsahem druhé kapitoly. Za tímto účelem definujeme tzv. „centrální cestu“ jako křivku, jejíž body porušují nelineární KKT podmínku a dokážeme, že taková křivka existuje a je definicí určena jednoznačně. Zavedeme pojem „centrující parametr“ $\sigma \in \langle 0, 1\rangle$, „míra duality“ $\mu \geq 0$ a formálně popíšeme základní schéma primárně–duálních metod. Ve druhé kapitole se také krátce zmíníme o logaritmických barierových metodách.

Ve třetí kapitole zavedeme okolí centrální cesty $\mathcal{N}_2(\theta)$, $\mathcal{N}_\infty(\theta)$, $\mathcal{N}_{-\infty}(\gamma)$ pro hodnoty $\theta \in \langle 0, 1\rangle$, $\gamma \in \langle 0, 1)$ a popíšeme tři metody path–following. Ukážeme, že (za podmínky neprázdnosti množiny ostře přípustných řešení) konvergují ve všech třech případech

hodnoty parametru μ_k k nule a podrobněji rozebereme konvergenci posloupnosti iterací $\{(x, y, s)\}$ k primárně–duálnímu řešení (x^*, y^*, s^*) úlohy LP.

U všech těchto metod uvažujeme počáteční přiblížení (x^0, y^0, s^0) z množiny ostře přípustných řešení, což je teoretický předpoklad, kterého v praxi většinou nelze dosáhnout. V kapitole čtvrté se proto sosustředíme na nepřípustné metody vnitřního bodu, které tento předpoklad nesplňují. Podrobněji se soustředíme an jednu z těchto metod, pro niž na konci kapitoly opět uvedeme konvergenční větu.

Popis tří v praxi realizovaných metod včetně teoretické analýzy, konvergenčních vět a vět popisujících složitost algoritmů je obsahem přílohy C. V příloze A jsou velmi krátce uvedena schémata primárních metod a duálních metod, příloha B obsahuje technické důkazy vět z předcházejících kapitol, úpravu matice soustavy pro nalezení primárně–duálního řešení na jednodušší tvar a větu o její jednoznačné řešitelnosti. Na závěr jsou uvedeny definice jednotlivých typů konvergencí.

Kapitola 1

Úloha lineárního programování; vztahy mezi primární a duální úlohou

1.1 Formulace úlohy LP

Úlohou lineárního programování je nalézt minimum resp. maximum lineární funkce na množině vymezené lineárními podmínkami. Obvykle ji uvažujeme v jedné ze dvou uvedených formulací:

A. standardní formulace (standard form):

$$\min c^T x \quad \text{na mnocheckzinchecke} \quad \{x \in R^n : Ax = b,\ x \geq 0\}$$

B. formulace pomocí nerovností (inequality form)

$$\min c^T x \quad \text{na množinchecke} \quad \{x \in R^n : Ax \geq b,\ x \geq 0\}.$$

V obou případech předpokládáme matici A typu $m \times n$, $c \in R^n$, $b \in R^m$.
Obě uvedené formulace jsou ekvivalentní a každou úlohu lineárního programování lze pomocí jednoduchých úprav převést na libovolnou z nich. Volné proměnné x_j, tj. takové, které nemají omezení typu $x_j \geq 0$, nahradíme rozdílem $x_j = x'_j - x''_j$, kde $x'_j \geq 0$, $x''_j \geq 0$. Nerovnost $Ax \geq b$ převedeme na rovnost přidáním nových proměnných následujícím způsobem

$$\begin{aligned} Ax + \hat{x} &= b, \\ \hat{x} &\geq 0 \end{aligned}$$

a obráceně rovnost $Ax = b$, převedeme na nerovnosti úpravou

$$\begin{aligned} Ax &\leq b \\ \& \quad Ax &\geq b. \end{aligned}$$

Znaménko nerovnosti změníme vynásobením obou stran nerovnosti číslem -1. Konečně úloha hledání maxima funkcionálu $c^T x$ je ekvivalentní úloze hledání minima funkcionálu $-c^T x$.

Nebude-li řečeno jinak, uvažujeme úlohu lineárního programování v standardní formulaci a předpokládáme, že matice A má hodnost rovnou m.

Přípustným řešením úlohy lineárního programování rozumíme vektor $x \in R^n$, který splňuje podmínky

$$\begin{aligned} Ax &= b, \\ x &\geq 0. \end{aligned}$$

Funkci $f(x) = c^T x$ nazýváme účelovou nebo také cílovou funkcí úlohy lineárního programování.

Optimálním řešením úlohy lineárního programování je pak vektor $x^* \in R^n$ takový, že

1. $x^* \in R^n$ je přípustným řešením úlohy
2. $c^T x^* \leq c^T x$ pro každé přípustné řešení $x \in R^n$.

1.2 Dualita

Definujme nejprve pojem „primární úloha“ a „duální úloha“ a podívejme se na základní vztahy týkající se přípustnosti a řešitelnosti primární úlohy v závislosti na duální a obráceně. V následujících odstavcích pak dokažme nutnou a postačující podmínku pro řešitelnost těchto úloh a zaveďme pojem primárně–duálního řešení úlohy lineárního programování.

Uvažujme opět úlohu

$$\min \{c^T x;\ Ax = b,\ x \geq 0\} \qquad (\hat{\mathrm{P}})$$

a vytvořme pro $(\hat{P})$ novou úlohu se stejnou maticí A a stejnými vektory b a c, která ovšem bude mít tvar

$$\max \{b^T y;\ A^T y \leq c\} \qquad (\hat{\mathrm{D}}).$$

Je zřejmé, že $(\hat{D})$ je opět úlohou lineárního programování, tentokrát však formulovanou pomocí nerovností. Duální proměnná y je nyní vektor z množiny R^m. Abychom od sebe úlohy odlišili, nazveme $(\hat{P})$ primární a $(\hat{D})$ duální úlohou lineárního programování a budeme je chápat jako dvojici vzájemně svázaných úloh. Že je to skutečně možné nám ukáže následující analýza. Dříve než k ní přistoupíme, definujme však ještě některé pojmy týkající se duální úlohy.

Často je výhodné formulovat jak primární, tak i duální úlohu ve standardním tvaru. Zavádíme proto novou proměnnou $s \in R^n$, tzv. slack variable a místo dvojice úloh $(\hat{P})$, $(\hat{D})$ řešíme dvojici úloh

$$\min\{c^T x;\ \ Ax = b,\ x \geq 0\} \tag{P}$$

$$\max\{b^T y;\ \ A^T y + s = c,\ s\ \geq 0\}. \tag{D}$$

Podobně jako tomu bylo u primární úlohy, nazýváme přípustným řešením duální úlohy vektor $(y, s) \in R^m \times R^n$, pro který je $A^T y + s = c$ & $s \geq 0$, duální účelovou funkcí funkci $f(x) := b^T y$ a optimálním řešením duální úlohy vektor $(y^*, s^*) \in R^m \times R^n$ pro který platí:

1. $(y^*, s^*) \in R^m \times R^n$ je přípustným řešením úlohy
2. $b^T y^* \geq b^T y$ pro každé přípustné řešení $(y, s) \in R^m \times R^n$.

Označme

$$\Omega_P := \{x^* :\ x^* \text{ je optimální řešení primární úlohy}\}, \tag{1.1}$$
$$\Omega_D := \{(y^*, s^*) :\ (y^*, s^*) \text{ je optimální řešení duální úlohy}\}. \tag{1.2}$$

Poznámka 1 Uvažujeme-li primární úlohu ve tvaru

$$\min\{c^T x;\ Ax \geq b,\ x \geq 0\}, \tag{$\tilde{\text{P}}$}$$

má duální úloha tvar

$$\max\{b^T y;\ A^T y \leq c,\ y \geq 0\}. \tag{$\tilde{\text{D}}$}$$

Poznámka 2 Chápeme-li (D) jako primární úlohu, pak má duální úloha k ní tvar (P).

Mezi optimálními řešeními úloh (P) a (D) existuje jistá souvislost, kterou popisují následující věty a lemmata.

Lemma 1 *(slabá věta o dualitě) Jsou-li x resp. (y, s) libovolná přípustná řešení úloh (P) resp. (D), potom platí*

$$c^T x \geq b^T y. \tag{1.3}$$

Důkaz:

$$c^T x = x^T c = x^T A^T y + x^T s \geq x^T A^T y = (Ax)^T y = b^T y. \square$$

Jinými slovy, hodnota účelové funkce primární úlohy v bodě x je horním odhadem hodnoty účelové funkce duální úlohy v témže bodě, a naopak, hodnota účelové funkce duální úlohy v bodě x je dolním odhadem hodnoty účelové funkce primární úlohy.

Velmi důležitý je důsledek lemmatu 1. Existují-li totiž přípustná řešení x^* resp. (y^*, s^*) úloh (P) resp. (D) taková, že $c^T x^* = b^T y^*$, pak x^* je optimálním řešením úlohy (P) a (y^*, s^*) optimálním řešením úlohy (D). Otázkou zůstává existence takových přípustných řešení. Odpověď na ni dává následující věta.

Věta 1 *(o dualitě) Pro úlohy (P) a (D) platí jedna z následujících alternativ*

(i) (P) i (D) jsou přípustné a obě mají optimální řešení $x^* \in \Omega_P$, $(y^*, s^*) \in \Omega_D$.

(ii) (P) je nepřípustná a účelová funkce (D) je (shora) neomezená.

(iii) (D) je nepřípustná a účelová funkce (P) je (zdola) neomezená.

(iv) Obě úlohy (P) i (D) jsou nepřípustné.

Důkaz věty o dualitě je uveden v Dodatku B.

Důsledek 1 *Předpokládejme, že primární úloha (P) je přípustná. Pak je její účelová funkce omezená zdola na množině přípustných řešení tehdy a jen tehdy, je-li duální úloha (D) přípustná.*

Podobně předpokládejme, že je duální úloha přípustná. Pak je její účelová funkce omezená shora na množině přípustných řešení tehdy a jen tehdy, je-li přípustná primární úloha.

Dříve, než popíšeme ještě jeden vztah mezi optimálními řešeními úlohy lineárního programování a přípustnými řešeními úlohy k ní duální, zaveďme následující terminologii.

Řekneme, že vektor $x \in R^n$ je ostře přípustným řešením primární úlohy, jestliže

$$Ax = b \ \ \& \ \ x > 0.$$

Podobně řekneme, že vektor $(y, s) \in R^m \times R^n$ je ostře přípustným řešením duální úlohy, jestliže

$$A^T y + s \ = c \ \ \& \ \ s \ > 0.$$

Věta 2 *Předpokládejme, že jak primární tak duální úloha jsou přípustné. Pak platí: má-li duální úloha alespoň jedno ostře přípustné řešení, je množina* Ω_P *neprázdná a omezená. Podobně má-li primární úloha alespoň jedno ostře přípustné řešení,je množina*

$$\{s^* : \ (y^*, s^*) \in \Omega_D \quad \text{pro nějaké} \ \ y^* \in R^m\}$$

neprázdná a omezená.

Důkaz: Dokážeme pouze první část tvrzení. Označme $(\bar{y}, \bar{s})$ ostře přípustné řešení duální úlohy a označme $\hat{x}$ libovolné přípustné řešení primární úlohy. Víme, že platí

$$0 \leq c^T \hat{x} - b^T \bar{y} = \bar{s}^T \hat{x}. \tag{1.4}$$

Nyní uvažujme množinu T definovanou

$$T := \left\{ x : Ax = b, x \geq 0, c^T x \leq c^T \hat{x} \right\}.$$

Množina T je neprázdná ($\hat{x} \in T$) a uzavřená. Pro každé $x \in T$ navíc platí

$$\sum_{i=1}^{n} \bar{s}_i x_i = \bar{s}^T x = c^T x - b^T \bar{y} \leq c^T \hat{x} - b^T \bar{y} = \bar{s}^T \hat{x}.$$

Protože všechny sčítance na levé straně výrazu jsou nezáporné, je

$$x_i \leq \frac{1}{\bar{s}_i} \bar{s}^T \hat{x},$$

z čehož plyne

$$||x||_\infty \leq \left(\max_{1 \leq i \leq n} \frac{1}{\bar{s}_i} \right) \bar{s}^T \hat{x}. \tag{1.5}$$

Protože x byl libovolný bod z množiny T, můžeme říct, že T je omezená. Tedy na ní musí funkce $c^T x$ nabývat svého minima, což znamená, že existuje bod x^* takový, že

$$x^* \in T,$$
$$c^T x^* \leq c^T x \quad \text{pro všechna } x \in T.$$

Je zřejmé, že bod x^* náleží množině Ω_p. Ω_p je tedy neprázdná a — podle (1.5) — omezená. □

1.3 Farkasovo lemma, KKT podmínky a primárně–duální řešení

Farkasovo lemma je jedním z nejpopulárnějších tvrzení teorie lineárního programování a v literatuře ho můžeme nalézt v několika různých verzích. Zde uvedená verze pochází z článku D. Goldfarb, M.J. Todd [3]. Lemma lze odvodit na základě důsledku 1.

Věta 3 *(Farkasovo lemma)*
Systém

$$Ax = b, \;\; x \geq 0 \tag{I}$$

je neřešitelný právě tehdy, když je systém

$$A^T y \leq 0, \;\; b^T y > 0 \tag{II}$$

řešitelný.

Důkaz: Uvažujme dvojici úloh lineárního programování

$$\min\{0^T x; \; Ax = b, \; x \geq 0\} \tag{P_0}$$

$$\max\{b^T y;\ A^T y \leq 0\} \tag{D$_0$}$$

Jelikož (D$_0$) je přípustná (např. $y = 0$ je řešením), je (P$_0$) nepřípustná (neboli systém (I) je neřešitelný) právě když má (D$_0$) neomezenou účelovou funkci a to nastává právě tehdy, když je systém (II) řešitelný.

Důkaz této ekvivalence není obtížný, ale pro úplnost ho zde uveďme. Nechť existuje y tak, že $A^T y \leq 0$ a zároveň $b^T y > 0$ (tj. y nenulové), pak pro každé kladné α platí:

$$(\alpha y)^T A \ \leq 0$$

(tedy αy je přípustné pro (D$_0$)) a zároveň

$$b^T(\alpha y) > 0.$$

Čili

$$\lim_{\alpha \to \infty} b^T(\alpha y) = \infty.$$

Obráceně, nechť takové y neexistuje. Pro každé y pak platí buď

$$A^T y > 0 \quad \text{nebo} \quad b^T y \leq 0.$$

Je-li ovšem vektor y přípustný pro (D$_0$), není $A^T y > 0$ a nutně tedy musí být splněno $b^T y \leq 0$, což znamená omezenost účelové funkce (D$_0$). □

Věta 4 *(Complementary slackness) Uvažujme dvojici úloh (P) a (D) ve tvaru:*

$$\min\{c^T x;\ \ Ax \geq b,\ x \geq 0\}, \tag{$\tilde{\text{P}}$}$$

$$\max\{b^T y;\ \ A^T y\ \leq c,\ y \geq 0\}. \tag{$\tilde{\text{D}}$}$$

a nechť x resp. y jsou přípustná řešení ($\tilde{P}$) resp. ($\tilde{D}$). Pak x resp. y jsou optimální řešení ($\tilde{P}$) resp. ($\tilde{D}$) tehdy a jen tehdy, jestliže

$$(c - A^T y)^T x = 0 \tag{1.6}$$

a zároveň

$$y^T(Ax - b) = 0. \tag{1.7}$$

Důkaz: Pro x a y přípustná definujme

$$\begin{aligned} w &:= Ax - b \geq 0, \\ s &:= c - A^T y \geq 0. \end{aligned}$$

Z nerovností $(\tilde{P})$, $(\tilde{D})$ plyne

$$c^T x \geq y^T A x \geq y^T b. \tag{1.8}$$

Jsou-li splněny podmínky (1.6), (1.7), nastává rovnost a tedy x je optimálním řešením primární úlohy, y je optimálním řešením duální úlohy.

Naopak, je -li x optimálním řešením primární úlohy a y optimálním řešením duální úlohy, je podle slabé věty o dualitě, $c^T x = b^T y$. Ve vztahu (1.8) nastávají rovnosti, z čehož plyne (1.6), (1.7). □

Poznámka 3 Poněvadž $(w, x, s, y) \geq 0$, je možné podmínky (1.6) a (1.7) přepsat do tvaru

$$s_i = (c^T - A^T y)_i = 0 \quad \text{nebo} \quad x_i = 0 \quad \text{pro každé} \quad 1 \leq i \leq n,$$

$$w_i = (Ax - b)_i = 0 \quad \text{nebo} \quad y_i = 0 \quad \text{pro každé} \quad 1 \leq i \leq m.$$

Poznámka 4 Pro dvojici úloh

$$\min\{c^T x;\ Ax = b,\ x \geq 0\}, \tag{P}$$

$$\max\{b^T y;\ A^T y \leq c\} \tag{D}$$

je podmínka (1.7) triviálně splněna. Nutnou a postačující podmínkou optimality je tudíž pouze podmínka (1.6).

Věta 5 *Vektor x je optimálním řešením úlohy*

$$\min\{c^T x;\ Ax = b,\ x \geq 0\} \tag{P}$$

právě tehdy, když existují vektory $y \in R^m$, $s \in R^n$ takové, že

(i) $Ax = b,\ x \geq 0,$

(ii) $A^T y + s\ = c,\ s \geq 0,$

(iii) $s^T x = 0.$

Důkaz plyne z definice přípustnosti primární a duální úlohy a z věty 4.

Věta 6 *(Karushovy - Kuhnovy - Tuckerovy podmínky) Vektor $x^* \in R^n$ je řešením úlohy (P) právě tehdy, když existují vektory $y^* \in R^m$, $s^* \in R^n$ tak, že platí*

$$\begin{aligned} Ax &= b, && && (1.9)\\ A^T y + s &= c, && && (1.10)\\ s_i x_i &= 0, && i = 1, \ldots, n && (1.11)\\ (x, s) &\geq 0 && && (1.12) \end{aligned}$$

pro $(x, y, s) = (x^*, y^*, s^*)$.

Důkaz: Věta je přepisem podmínek z věty 5 pro úlohu lineárního programování, přičemž pro odvození vztahu (1.11) užíváme navíc vlastnosti uvedené v poznámce 3. □

Vztah (1.11) se nazývá podmínka komplementarity a neříká nic jiného, než že pro každý index i je alespoň jedna z komponent s_i, x_i nulová, tj. nenulové prvky obou vektorů se vyskytují pouze na doplňkových pozicích.

Právě uvedené tvrzení charakterizuje vztah mezi primární a duální úlohou. Formálně můžeme uvést duální podmínku optimality v následujícím tvaru:

Věta 7 *Vektor $(y^*, s^*) \in R^m \times R^n$ je řešením úlohy (D) právě tehdy, když existuje vektor $x^* \in R^n$ tak, že pro $(x, y, s) = (x^*, y^*, s^*)$ platí podmínky (1.9)–(1.12).*

Srovnáme-li podmínky (1.9)–(1.12) z pohledu řešitelnosti primární úlohy a z pohledu řešitelnosti duální úlohy, lze závěrem říci, že vektor (x^*, y^*, s^*) je řešením systému (1.9)–(1.12) právě tehdy, když x^* řeší primární úlohu a (y^*, s^*) úlohu duální. Vektor (x^*, y^*, s^*) se potom nazývá primárně–duální řešení úlohy lineárního programování.

Na závěr zaveďme ještě označení

$$\Omega := \Omega_P \times \Omega_D = \{(x^*, y^*, s^*) : \ (x^*, y^*, s^*) \quad \text{splňují podmínky (1.9)} - -\text{(1.12)}\}$$

Zobecněný tvar KKT podmínek pro obecně nelineární úlohu a rozšíření Farkasova lemmatu je uveden na konci této kapitoly.

1.4 Rozklad množiny indexů a striktní komplementarita

Je-li (x^*, y^*, s^*) primárně–duální řešení úlohy lineárního programování, pak z podmínky (1.11) víme, že pro každý index $i \in \{1, \ldots, n\}$ je buď x_i nebo s_i rovné nule. Na základě tohoto poznatku definujme dvě podmnožiny množiny indexů $\{1, \ldots, n\}$

$$\mathcal{B} = \{j \in \{1, \ldots, n\} : x_j^* \neq 0 \text{ pro nějaké } x^* \in \Omega_P\}, \tag{1.13}$$

$$\mathcal{N} = \{j \in \{1, \ldots, n\} : s_j^* \neq 0 \text{ pro nějaké } (y^*, s^*) \in \Omega_D\}. \tag{1.14}$$

Ve skutečnosti jsou množiny $\mathcal{B}$ a $\mathcal{N}$ disjunktní; každý index $j \in \{1, \ldots, n\}$ leží buď v $\mathcal{N}$ nebo v $\mathcal{B}$, ale nikdy ne v obou množinách zároveň. Tento fakt není obtížné dokázat. Kdyby totiž některý index j náležel jak množině $\mathcal{N}$, tak množině $\mathcal{B}$, existovalo by primárně–duální řešení (x^*, y^*, s^*), pro nějž je $x_j^* > 0$ a zároveň $s_j^* > 0$, potom ale i součin $x_j^* s_j^* > 0$, což odporuje podmínce (1.11).

Skutečnost, že $\mathcal{B} \cup \mathcal{N} = \{1, \ldots, n\}$ je známá jako Goldmanova–Tuckerova věta. Větu vyslovíme nyní a její důkaz uvedeme na konci kapitoly.

Věta 8 *(Goldmanova–Tuckerova)* $\mathcal{B} \cup \mathcal{N} = \{1, \ldots, n\}$. *To znamená, že existuje alespoň jedno primární řešení* $x^* \in \Omega_P$ *a alespoň jedno duální řešení* $(y^*, s^*) \in \Omega_D$ *takové, že* $x^* + s^* > 0$.

Primárně–duální řešení (x^*, y^*, s^*), pro něž je $x^* + s^* > 0$, se obvykle nazývají striktně komplementární řešení. Věta 8 zaručuje existenci alespoň jednoho takového řešení. Jsou ovšem úlohy lineárního programování, které mají vícenásobná řešení, z nichž některá jsou striktně komplementární a jiná nejsou. Jako příklad můžeme uvést úlohu

$$\min x_1 \quad \text{na množině} \quad \{x : \ x_1 + x_2 + x_3 = 1, \ x \geq 0\},$$

jejímž primárně–duálním řešením je každý vektor tvaru (x^*, y^*, s^*), kde

$$x^* = (0, t, 1-t), \quad y^* = 0, \quad s^* = (1, 0, 0); \quad t \in \langle 0, 1 \rangle.$$

1.5 Zobecnění Farkasova lemmatu a KKT podmínek

Uvažujme obecný optimalizační problém s lineárními omezujícími podmínkami, který má tvar

$$\min f(u) \quad \text{na množině} \quad \{u \ \in U : \ Cu = d, \ Gu \geq h\}, \tag{1.15}$$

kde f je hladká funkce, $U \in R^N$ je otevřená množina, C a G jsou matice a d a h jsou vektory.

Řekneme, že $u^* \in R^N$ je lokální řešení úlohy (1.15), jestliže

1. $u^* \in R^n$ je přípustným řešením úlohy, tj. $u^* \in U$, $Cu^* = d$ a $Gu^* \geq h$,
2. existuje číslo ρ tak, že $f(u^*) \leq f(u)$ pro každé přípustné řešení u, pro nějž $\| u \ - u^* \| < \rho$.

Bod u^* je ostré lokální řešení, jestliže je lokální řešení a navíc pro každé přípustné řešení u, pro nějž $\| u \ - u^* \| < \rho$ & $u \neq u^*$ platí $f(u^*) < f(u)$.

Definice je možné rozšířit na pojem globálního resp. ostrého globálního řešení.

Věta 9 *(Farkasovo lemma) Pro každou matici* $G \in R^{P \times N}$ *a každý vektor* $g \in R^N$ *platí buď*

I. $Gt \geq 0$, $g^T t < 0$ *má řešení* $t \in R^N$
nebo

II. $G^T s = g$, $s \geq 0$ *má řešení* $s \in R^P$,
ale nikdy obojí současně.

Důsledek 2 *Pro každou dvojici matic* $G \in R^{P \times N}$ *a* $H \in R^{Q \times N}$ *a každý vektor* $g \in R^N$ *platí buď*

I. $Gt \geq 0$, $Ht = 0$, $g^T t < 0$ *má řešení* $t \in R^N$
nebo

II. $G^T s + H^T r = g$, $s \geq 0$ *má řešení* $s \in R^P$, $r \in R^Q$,
ale nikdy obojí současně.

Věta 10 *(KKT podmínky) Nechť* u^* *je lokální řešení úlohy (1.15) a nechť* f *je diferencovatelná v jistém okolí bodu* u^**. Pak existují vektory* y *a* z *tak, že platí*

$$\begin{aligned} Cu^* &= d, && (1.16)\\ \nabla f(u^*) - C^T y - G^T z &= 0, && (1.17)\\ Gu^* &\geq h, && (1.18)\\ z &\geq 0, && (1.19)\\ z^T(Gu^* - h) &= 0. && (1.20) \end{aligned}$$

Vektory y *a* z *jsou Lagrangeovy multiplikátory.*

Na základě zobecněných KKT podmínek můžeme zobecnit také definici striktně komplementárního řešení.

Definice Řešení u^* problému (1.15) a jemu odpovídající Lagrangeovy multiplikátory (y, z) jsou striktně komplementární, jestliže $z + (Gu^* - h) > 0$.

Na základě podmínek (1.18), (1.19) a (1.20) můžeme pak říct, že u striktně komplementárních řešení (u^*, y, z) platí pro každý index i buď

$$z_i = 0 \ \ \& \ \ (Gu^* - h)_i > 0$$

nebo

$$z_i > 0 \ \ \& \ \ (Gu^* - h)_i = 0.$$

Z věty 10 vyplývá, že KKT podmínky jsou nutné podmínky optimality: je-li u^* lokálním řešením úlohy (1.15), pak existují vektory (y, z) tak, že platí (1.16)–(1.20). V následujícím odstavci ukážeme, že za určitých dalších předpokladů jsou také podmínkami postačujícími.

1.6 Konvexita a globální řešení

Je-li U otevřená konvexní množina (U může být i celý prostor R^N), pak je konvexní i množina přípustných řešení pro úlohu (1.15). Je-li navíc účelová funkce f konvexní funkcí na množině přípustných řešení, nazýváme úlohu (1.15) úlohou konvexního programování.

Věta 11 *(i) Nechť (1.15) je úlohou konvexního programování. Existují-li vektory* y *a* z *tak, že trojice* (u^*, y, z) *splňuje KKT podmínky (1.16)–(1.20), pak je* u^* *globálním řešením úlohy (1.15).*

Důkaz: Nechť je dán vektor (u^*, y, z), který splňuje podmínky (1.16)–(1.20). Naší snahou bude dokázat, že

$$f(u^* + v) \geq f(u^*) \tag{1.21}$$

pro každé v takové, že $u^* + v$ je přípustné řešení úlohy (1.15). Protože f je konvexní, platí pro každý vektor v, pro nějž $u^* + v \ \in U$:

$$f(u^* + \alpha v) \leq (1 - \alpha)\, f(u^*) + \alpha f(u^* + v)$$

a tedy

$$\frac{1}{\alpha}\left(f(u^* + \alpha v) - f(u^*)\right) \leq f(u^* + v) - f(u^*).$$

Využijme teď toho, že f je diferencovatelná a proveďme limitu pro $\alpha \to 0$. Dostaneme vztah

$$f(u^* + v) \geq f(u^*) + v^T \nabla f(u^*). \tag{1.22}$$

Protože je množina přípustných řešení konvexní, můžeme každé přípustné řešení vyjádřit jako $u^* + v$ pro vhodné v. Poněvadž $Cu^* = d$ a $C\,(u^* + v\ ) = d$, je nutně $Cv\ = 0$. Rozdělme nyní, na základě podmínky (1.18), množinu indexů následovně. V množině I_A nechť jsou „aktivní“ indexy, tj. takové, pro něž

$$(Gu^*)_i = h_i$$

a v množině I_N nechť jsou „neaktivní“ indexy, tj. takové, pro něž

$$(Gu^*)_i > h_i.$$

Nechť nejprve $i \in I_A$. Protože $u^* + v$ je přípustné, musí platit $(Gv\ )_i \geq 0$. Je-li $i \in I_N$, musí, podle (1.19) a (1.20), být $z_i = 0$. Z (1.17) získáme

$$\begin{aligned} v^T \nabla f(u^*) &= v^T \left(C^T y + G^T z\ \right) = y^T C v\ + \sum_{i \in I_A} z_i\,(Gv\ )_i + \sum_{i \in I_N} z_i\,(Gv\ )_i \\ &= \sum_{i \in I_A} z_i\,(Gv\ )_i \geq 0. \end{aligned}$$

Po dosazení $v^T \nabla f(u^*)$ do pravé strany výrazu (1.22) získáme vztah (1.21). □

1.7 Důkaz Goldmanovy–Tuckerovy věty

Na závěr kapitoly uveďme ještě důkaz věty 8. Využijeme důsledku 2 Farkasova lemmatu.

Buď $\mathcal{J}$ množina indexů $j \in \{1, \ldots, n\}$, které nenáleží ani do $\mathcal{B}$, ani do $\mathcal{N}$. Dokážeme, že množina $\mathcal{J}$ je prázdná.

Už jsme ukázali, že $\mathcal{B} \cap \mathcal{N} = \emptyset$; $\mathcal{B} \cup \mathcal{N} \cup \mathcal{J}$ je tedy rozklad množiny $\{1, \ldots, n\}$. Označme A_j j–tý sloupec matice A a $A_\mathcal{B}$ resp. $A_\mathcal{J}$ submatice obsahující sloupce A_j, kde $j \in \mathcal{B}$ resp. $j \in \mathcal{J}$.

Zvolme libovolné $j \in \mathcal{J}$. Dokážeme, že $j \in \mathcal{N}$ nebo $j \in \mathcal{B}$ v závislosti na tom, zda existuje w, pro nějž

$$\begin{array}{rcl} A_j^T w & < & 0, \\ -A_k^T w & \geq & 0 \quad \text{pro všechna } k \ \in \mathcal{J} - \{j\}, \\ A_\mathcal{B}^T w & = & 0. \end{array} \tag{1.23}$$

Předpokládejme nejprve, že takové w existuje.

Buď (x^*, y^*, s^*) primárně–duální řešení, pro nějž $s^*_\mathcal{N} > 0$ a definujme vektor $(\bar{y}, \bar{s})$ jako

$$\begin{array}{rcl} \bar{y} & := & y^* + \varepsilon w, \\ \bar{s} & := & c - A^T \bar{y} = s^* - \varepsilon A^T w, \end{array}$$

přičemž ε zvolme takové, aby

$$\begin{array}{rcl} \bar{s}_j & = & s^*_j - \varepsilon A_j^T w > 0, \\ \bar{s}_k & = & s^*_k - \varepsilon A_j^T w \geq 0, \quad \forall k \ \in \mathcal{J} - \{j\}, \\ \bar{s}_\mathcal{B} & = & s^*_\mathcal{B} = 0, \\ \bar{s}_\mathcal{N} & = & s^*_\mathcal{N} - \varepsilon A_\mathcal{N}^T w > 0. \end{array}$$

Z těchto vztahů vyplývá, že $(\bar{y}, \bar{s})$ je přípustným řešením duální úlohy. Ve skutečnosti je optimálním řešením, neboť pro každé primární řešení x^* musí být $x^*_\mathcal{N} = 0$ a tedy $\bar{s}^T x^* = 0$. Podle definice (1.14) musí $j \in \mathcal{N}$.

Teď předpokládejme, že takové w naopak neexistuje. Podle důsledku Farkasova lemmatu musí mít systém

$$-\sum_{k \in \mathcal{J} - \{j\}} s_k A_k + A_\mathcal{B} r = A_j, \qquad s_k \geq 0 \tag{1.24}$$

řešení pro každé $k \ \in \mathcal{J} - \{j\}$. Definujeme-li vektor $v \in R^{|\mathcal{J}|}$ jako

$$v_j := 1, \qquad v_k := s_k \qquad (k \ \in \mathcal{J} - \{j\}),$$

můžeme (1.24) přepsat jako

$$\begin{array}{rcl} A_\mathcal{J} v & = & A_\mathcal{B} r, \\ v & \geq & 0, \\ v_j & \geq & 0. \end{array} \tag{1.25}$$

Nyní buď x^* primární řešení, pro nějž $x^*_{\mathcal{B}} > 0$ a definujme $\bar{x}$ jako

$$\bar{x}_{\mathcal{B}} := x^*_{\mathcal{B}} - \varepsilon r, \qquad \bar{x}_{\mathcal{J}} := \varepsilon v, \qquad \bar{x}_{\mathcal{N}} := 0.$$

Po dosazení do (1.25) získáme $A\bar{x} = b$ a pro dostatečně malá ε navíc $\bar{x} \geq 0$. Tedy $\bar{x}$ je přípustné. Ve skutečnosti je $\bar{x}$ optimální, protože $\bar{x}_{\mathcal{N}} = 0$. Protože $v_j = 1$, platí také $\bar{x}_{\mathcal{J}} = \varepsilon > 0$ a tedy $j \in \mathcal{B}$ podle (1.13).

Dokázali jsme, že každý index $j \in \mathcal{J}$ náleží buď $\mathcal{B}$ nebo $\mathcal{N}$ a, podle definice $\mathcal{J}$, je tedy $\mathcal{J} = \emptyset$. Důkaz je hotov. □

Kapitola 2

Primárně–duální metody

V předchozí kapitole jsme popsali primárně–duální řešení dvojice úloh

$$\min\{c^T x;\ Ax = b,\ x \geq 0\} \tag{P}$$

$$\max\{b^T y;\ A^T y + s = c,\ s\ \geq 0\} \tag{D}$$

a dokázali jsme, KKT podmínky (1.9) - (1.12) jsou nutné a postačující pro to, aby vektor (x^*, y^*, s^*) byl primárně–duálním řešením úlohy lineárního programování.

Přepišme tyto podmínky v poněkud odlišném tvaru jako

$$F(x,y,s) := \begin{bmatrix} Ax - b \\ A^T y + s\ - c \\ XSe \end{bmatrix} = 0, \tag{2.1}$$

$$(x,s) \geq 0, \tag{2.2}$$

kde $X = \mathrm{diag}(x_1, \ldots, x_n)$, $S = \mathrm{diag}(s_1, \ldots, s_n)$, $e = (1, 1, \ldots, 1)^T$.

Všimněme si, že funkce $F : R^{2n+m} \to R^{2n+m}$ je nelineární pouze v posledních n složkách.

Primárně duální metody hledají primárně–duální řešení (x^*, y^*, s^*) modifikovanou Newtonovou metodou, aplikovanou na soustavu (2.1). Směr a délka kroku jsou voleny tak, aby v každé iteraci platila podmínka (2.2) ostře, tj. $(x^k, s^k) > 0$ pro každé k. Tato vlastnost je důvodem pro název „vnitřní bod“. Respektujeme-li uvedené podmínky, vyhneme se při výpočtu bodům, pro něž je sice $F(x, y, s) = 0$, ale už ne $(x, s) \geq 0$. Takových bodů je mnoho, ale žádný z nich v sobě nenese jakoukoli pro nás užitečnou informaci o úlohách (P) a (D).

Většina metod vnitřního bodu požaduje, aby byly všechny iterace ostře přípustné. Definujeme-li množinu přípustných řešení $\mathcal{F}$ a množinu ostře přípustných řešení $\mathcal{F}^0$ předpisy

$$\begin{aligned} \mathcal{F} &:= \{(x,y,s);\ Ax = b,\ A^T y + s\ = c,\ (x,s) \geq 0\} \\ \mathcal{F}^0 &:= \{(x,y,s);\ Ax = b,\ A^T y + s\ = c,\ (x,s) > 0\}, \end{aligned}$$

můžeme tuto podmínku přepsat ve tvaru

$$(x^k, y^k, s^k) \in \mathcal{F}^0 \qquad \forall k.$$

Jako většina iteračních optimalizačních procesů, mají i metody vnitřního bodu dvě základní části: proceduru na vytvoření iteračního kroku a parametr míry očekávaného přiblížení (measure of the desirability) v každém bodě přípustné oblasti. Jak už jsme uvedli, výběr směru je založen na Newtonově metodě pro řešení nelineární soustavy (2.1). Newtonova metoda lineárně aproximuje funkci F v okolí daného bodu a směr $(\Delta x, \Delta y, \Delta s)$ získává vyřešením soustavy lineárních rovnic

$$J(x, y, s) \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = -F(x, y, s),$$

kde J je Jacobiho matice funkce F. Je-li daný bod (x, y, s) ostře přípustný (čili $(x, y, s) \in \mathcal{F}^0$), má Newtonova soustava tvar

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -XSe \end{bmatrix}. \tag{2.3}$$

Jednotkový krok v tomto směru ale většinou nemůžeme provést, neboť bychom porušili podmínku $(x, s) \geq 0$. Proto volíme v každé iteraci navíc délku kroku $\alpha^k \in \langle 0, 1 \rangle$. Nové přiblížení má potom tvar

$$(x, y, s) + \alpha^k(\Delta x, \Delta y, \Delta s)$$

Často je, bohužel, možné volit α^k pouze velmi malé ($\ll 1$) a metoda, která vybírá směr na základě (2.3) konverguje velmi pomalu.

Primárně–duální metody proto modifikují základní Newtonův algoritmus, a to dvěma způsoby:

1. Odkloní směr $(\Delta x, \Delta y, \Delta s)$ dovnitř nezáporného orthantu $(x, s) \geq 0$. V tomto odkloněném směru pak můžeme provést delší krok, aniž bychom podmínku $(x, s) > 0$ porušili.

2. Zachovají hodnoty složek vektoru (x, s) dostatečně daleko od hranice nezáporného orthantu, čímž předcházejí případnému zdegenerování metody a urychlují konvergenci.

2.1 Centrální cesta

Centrální cesta $\mathcal{C}$ je křivka tvořená ostře přípustnými řešeními, která hraje v teorii primárně–duálních algoritmů velmi důležitou roli. Je parametrizovaná skalárním parametrem $\tau > 0$ a každý bod $(x_\tau, y_\tau, s_\tau) \in \mathcal{C}$ řeší následující soustavu

$$
\begin{aligned}
Ax &= b, && (2.4)\\
A^T y + s &= c, && (2.5)\\
x_i s_i &= \tau, \quad i = 1, \ldots, n && (2.6)\\
(x, s) &> 0. && (2.7)
\end{aligned}
$$

Tyto podmínky se od podmínek KKT liší pouze v pravé straně výrazu (2.6), kde namísto podmínky komplementarity (1.11) požadujeme, aby součin $x_i s_i$ nabýval stejných hodnot pro všechna i. Na základě (2.4)–(2.7) můžeme centrální cestu definovat jako

$$\mathcal{C} := \{(x_\tau, y_\tau, s_\tau);\ \ \tau > 0\}$$

Jiný způsob popisu je užít značení zavedené v (2.1), (2.2) a psát

$$F(x_\tau, y_\tau, s_\tau) = \begin{bmatrix} 0 \\ 0 \\ \tau e \end{bmatrix} \tag{2.8}$$

$$(x_\tau, s_\tau) > 0$$

Soustava (2.4)–(2.7) aproximuje soustavu (1.9)–(1.12) a to tím lépe, čím je parametr τ blíže k nule. Jestliže tedy pro $\tau \to 0$ konverguje $\mathcal{C}$ k nějakému bodu, pak je tento bod primárně–duálním řešením úlohy lineárního programování.

Většina primárně–duálních metod užívá Newtonovu metodu nikoli přímo pro řešení soustavy $F(x, y, s) = 0$, ale pro nalezení bodů, ležících na centrální cestě $\mathcal{C}$. Metoda je ve skutečnosti složena ze dvou iteračních cyklů. Ve vnějším cyklu volíme hodnotu τ a to tak, aby $\tau \to 0$ a ve vnitřním cyklu řešíme soustavu (2.4)–(2.7). Tím (pro pevné τ) získáme modifikovaný směr $(\Delta x, \Delta y, \Delta s)$. V tomto směru pak můžeme provést delší krok.

Abychom tuto modifikaci mohli popsat lépe, zavedeme nyní dva nové pojmy. centrující parametr $\sigma \in \langle 0, 1\rangle$ (centering parametr) a míru duality μ ($\mu \geq 0$) (duality measure), definovanou vztahem

$$\mu := (1/n) \sum_{i=1}^{n} x_i s_i = (x^T s)/n \tag{2.9}$$

která udává průměrnou hodnotu součinů $x_i s_i$.

V každém vnějším kroku zvolíme hodnotu $\tau = \sigma\mu$; ve vnitřním pak řešíme soustavu pro výběr směru, která má tvar

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -XSe + \sigma\mu e \end{bmatrix} \tag{2.10}$$

Tím provedeme jeden krok Newtonovy metody směrem k bodu $(x_{\sigma\mu}, y_{\sigma\mu}, s_{\sigma\mu}) \in \mathcal{C}$, pro který $x_i s_i = \sigma\mu$.

Je-li $\sigma = 1$, rovnice (2.10) definují tzv. centrující směr (centering direction), tj. jeden krok Newtonovy metody směrem k bodu $(x_\mu, y_\mu, s_\mu) \in \mathcal{C}$, pro který $x_i s_i = \mu$.

2.2 Primárně–duální schéma

Na základě principů, uvedených v předchozích odstavcích, můžeme vytvořit obecné schéma primárně–duálních metod.

PD schéma
Dáno $(x^0, y^0, s^0) \in \mathcal{F}^0$
for $k = 0, 1, 2, \ldots$
zvolme $\sigma \in \langle 0, 1 \rangle$,
položme $\mu_k := ((x^k)^T s^k)/n$
a řešme soustavu

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} \Delta x^k \\ \Delta y^k \\ \Delta s^k \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -X^k S^k e + \sigma_k \mu_k e \end{bmatrix} \tag{2.11}$$

definujme

$$(x^{k+1}, y^{k+1}, s^{k+1}) := (x^k, y^k, s^k) + \alpha^k (\Delta x^k, \Delta y^k, \Delta s^k) \tag{2.12}$$

kde α^k volíme tak, aby $(x^{k+1}, s^{k+1}) > 0$.
end (for).

2.3 Logaritmické barierové metody

V tomto odstavci bych ráda v krátkosti uvedla poněkud odlišný přístup k metodám vnitřního bodu a jiné odvození soustavy rovnic, definující centrální cestu.

Uvažujme nejprve obecný problém tvaru

$$\min f(x) \tag{2.13}$$

$$\text{na množině } \{x : c_i(x) \geq 0, \;\; 1 \leq i \leq m\}, \tag{2.14}$$

kde $x = (x_1, \ldots, x_n)^T$.

Základní filozofie spočívá v minimalizaci nově vytvořené funkce

$$B(x, \tau) := f(x) - \tau \sum_{i=1}^{n} \log(c_i(x)), \tag{2.15}$$

kde x je proměnná a τ je parametr. Tato funkce nabývá velkých hodnot jednak pro taková x, pro něž je velká hodnota funkce $f(x)$, ale také pro taková x, která jsou

příliš blízko hranice oblasti vymezené podmínkami (2.14). Opět tedy můžeme mluvit o metodách vnitřního bodu. Funkce $B(x,\tau)$ se obvykle nazývá logaritmická barierová funkce.

Problém minimalizace $B(x,\tau)$ řešíme iteračně. Iterační cyklus startujeme z bodu $\tau_0 > 0$, x^0 a v každém kroku hledáme

$$x^k := \arg\min B(x,\tau_k).$$

Následující hodnotu parametru pak volíme tak, aby

$$0 < \tau_{k+1} < \tau_k$$

a proces opakujeme, přičemž počátečním přiblížením pro nalezení bodu x^{k+1} je nyní bod x^k. Fiacco a McCormic [4] dokázali, že za určitých podmínek na funkce $f(x)$ a $c_i(x)$ konverguje (pro $\tau \to 0$) posloupnost iterací $\{x^k\}$ k přesnému řešení x^* úlohy (2.13), (2.14).

Pro naši úlohu lineárního programování ve tvaru

$$\min c^T x \quad \text{na množině} \quad \{x \in R^n : Ax = b, x \geq 0\}, \tag{2.16}$$

jíž odpovídá duální úloha

$$\max b^T y \quad \text{na množině} \quad \{(y,s) \in R^m \times R^n : \ A^T y + s = c, s \geq 0\} \tag{2.17}$$

má funkce $B(x,\tau)$ tvar

$$B(x,\tau) := c^T x - \tau \sum \log x_i. \tag{2.18}$$

Úlohu (2.16) převedeme na úlohu minimalizovat posloupnost funkcí $B(x,\tau)$ za podmínky $Ax = b$ a na tuto novou úlohu aplikujeme metodu Lagrangeových multiplikátorů. Lagrangeova funkce má (pro pevné τ)tvar

$$L(x,y,\tau) := c^T x - \tau \sum \log x_i - y^T(Ax - b) \tag{2.19}$$

a podmínky prvního řádu pro (2.19) potom jsou

$$\begin{aligned} c - \tau X^{-1} e - A^T y &= 0, \\ Ax &= b, \end{aligned} \tag{2.20}$$

kde X je opět diagonální matice, jejíž diagonální prvky tvoří složky vektoru x a $e = (1,\ldots,1)^T$. Položíme-li (podle (2.17)) $s := c - A^T y = \tau X^{-1} e$, můžeme podmínky (2.20) přepsat do tvaru

$$XSe = \tau e, \tag{2.21}$$

$$Ax - b = 0, \tag{2.22}$$

$$A^T y + s - c = 0. \tag{2.23}$$

kde $S = \mathrm{diag}(s_1, \ldots, s_n)$. Uvědomme si, že z definice funkce $B(x, \tau)$ implicitně vyplývá podmínka $x > 0$ a poněvadž pro každé i je $s_i = \tau / x_i$, platí také $s > 0$. Soustava (2.21)–(2.23) pak odpovídá soustavě (2.4)–(2.7).

Získaný systém nelineárních rovnic (2.21)–(2.23) opět řešíme Newtonovou metodou. Newtonova soustava pro směr $(\Delta x, \Delta y, \Delta s)$ má tvar

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} b - Ax \\ c - A^T y - s \\ \tau e - XSe \end{bmatrix}, \tag{2.24}$$

kde $\tau := \sigma(x^T s / n)$ a $\sigma \in \langle 0, 1 \rangle$. Nové přiblížení je potom

$$\begin{aligned} \hat{x} &:= x + \alpha_P \Delta x \\ \hat{y} &:= y + \alpha_D \Delta y \\ \hat{s} &:= s + \alpha_D \Delta s. \end{aligned}$$

Délka primárního kroku α_P a délka duálního kroku α_D je taková, aby $(\hat{x}, \hat{s}) > 0$. Algoritmus skončí, je-li hodnota $x^T s$ menší než předem pevně zvolené číslo ϵ.

Podrobnosti o metodách založených ma minimalizaci logaritmické barierové funkce lze nalézt např. v článcích M.H. Wright [5], J.Ji, Y.Ye [6], B.Jansen & C.Ross & T.Terlaky & J.P.Vial [7] nebo D.F.Shanno & E.M.Simantiraki [8].

2.4 Důkaz existence centrální cesty

Cílem tohoto odstavce je dokázat, že za podmínky neprázdnosti množiny $\mathcal{F}^0$ existuje pro každé $\tau > 0$ jednoznačné řešení soustavy (2.4)–(2.7) a tedy existuje (jednoznačně definovaná) centrální cesta.

Lemma 2 *Předpokládejme, že $\mathcal{F}^0 \neq \emptyset$. Pak pro každé $K \geq 0$ je množina*

$$\left\{ (x, s) : (x, y, s) \in \mathcal{F} \text{ pro nějaké } y, x^T s \leq K \right\}$$

omezená.

Důkaz: Buď $(\bar{x}, \bar{y}, \bar{s})$ libovolný vektor z $\mathcal{F}^0$ a (x, y, s) libovolný vektor z $\mathcal{F}$, pro který $x^T s \leq K$. Poněvadž $A\bar{x} = b$ a zároveň $Ax = b$, je $A(\bar{x} - x) = 0$. Podobně $A^T(\bar{y} - y) + (\bar{s} - s) = 0$. Z těchto dvou rovností vyplývá, že

$$(\bar{x} - x)^T (\bar{s} - s) = -(\bar{x} - x)^T A^T (\bar{y} - y) = 0.$$

Užijeme-li podmínku $x^T s \leq K$, získáme po úpravě

$$\bar{x}^T s + \bar{s}^T x \leq K + \bar{x}^T \bar{s}. \tag{2.25}$$

Definujme nyní ξ takto

$$\xi := \min_{1 \le i \le n} \left(\min(\bar{x}_i, \bar{s}_i) \right).$$

Protože $(\bar{x}, \bar{s}) > 0$, je $\xi > 0$. Po dosazení do (2.25) dostáváme

$$\xi e^T (x + s\ ) \le K\ + \bar{x}^T \bar{s},$$

což znamená

$$0 \le x_i \le \frac{1}{\xi} \left(K\ + \bar{x}^T \bar{s} \right), \qquad 0 \le s_i \le \frac{1}{\xi} \left(K\ + \bar{x}^T \bar{s} \right), \qquad 1 \le i \le n. \square$$

Definujme nyní množinu $\mathcal{H}^0$ následujícím způsobem

$$\mathcal{H}^0 := \left\{ (x, s) : (x, y, s) \in \mathcal{F}^0 \text{ pro nějaké } y \in R^m \right\}$$

a označme $(x_\tau, s_\tau) := \underset{(x,s) \in \mathcal{H}^0}{\operatorname{argmin}}\ f_\tau(x, s)$, kde f_τ je bariérová funkce definovaná předpisem

$$f_\tau(x, s) := \frac{1}{\tau} x^T s\ - \sum_{j=1}^{n} \log (x_j s_j).$$

Známe-li (x_τ, s_τ), víme z definice $\mathcal{H}^0$, že existuje $y_\tau \in R^m$ tak, že (x_τ, y_τ, s_τ) řeší soustavu (2.4) – (2.7). Na rozdíl od x a s není y_τ definované jednoznačně v případě, že matice A nemá plnou hodnost.

Funkce $f_\tau(x, s)$ se blíží k nekonečnu pro taková (x, s), pro něž se $x_j s_j$ blíží k nule. Nutně tedy $(x_\tau, s_\tau) > 0$.
Další vlastnosti funkce f:

1. *funkce f_τ je ostře konvexní na množině $\mathcal{H}^0$.*
   Je-li $\bar{x}$ libovolný vektor, pro který $A\bar{x} = b$, pak pro každé $(x, s) \in \mathcal{H}^0$ je

   $$x^T s\ = c^T x - b^T y = c^T x - \bar{x}^T A^T y = c^T x - \bar{x}^T (c - s\ ) = c^T x + \bar{x}^T s\ - \bar{x}^T c.$$

   Z čehož je zřejmé, že restrikce $x^T s$ na množinu $\mathcal{H}^0$ je ve skutečnosti lineární (a tedy konvexní) funkce. Druhý výraz z definice f_τ má pro každé $(x, s) > 0$ kladný Hessián a je tedy na $\mathcal{H}^0$ ostře konvexní. Celá funkce f_τ je potom ostře konvexní na $\mathcal{H}^0$.

2. *f_τ je na $\mathcal{H}^0$ zdola omezená.*
   Abychom ověřili tuto skutečnost, přepišme f_τ jako

   $$f_\tau(x, s) := \sum_{j=1}^{n} g\left(\frac{x_j s_j}{\tau} \right) + n - n \log \tau, \tag{2.26}$$

   kde

$$g(t) := t - \log t - 1.$$

Je zřejmé, že

(a) $g(t)$ je ostře konvexní na $(0, \infty)$

(b) $g(t) \geq 0$ pro $t \in (0, \infty)$, přičemž rovnosti se nabývá pouze pro $t = 1$

(c) $\lim_{t \to 0} g(t) = \lim_{t \to \infty} g(t) = \infty$.

Užijeme-li vlastnosti (b) ve výrazu (2.26), získáme

$$f_\tau(x, s) \geq n\,(1 - \log \tau)\,.$$

3. *Máme-li pevně dáno $\tau > 0$ a libovolné číslo K, potom všechny body (x, y), které náležejí množině*

$$\mathcal{L}_K := \left\{ (x, s) \in \mathcal{H}^0 : f_\tau(x, s) \leq K \right\}$$

splňují podmínky

$$x_i \in \langle M_l, M_u \rangle \ , \ s_i \in \langle M_l, M_u \rangle \qquad 1 \leq i \leq n \tag{2.27}$$

pro nějaká kladná čísla M_l a M_u.

Podle (2.26) je $f_\tau(x, s) \leq K$ tehdy a jen tehdy, je-li $\sum_{j=1}^{n} g\left(\frac{x_j s_j}{\tau}\right) \leq \bar{K}$, kde $\bar{K} = K \ - n + n \log \tau$. Zvolíme-li libovolný index i, dostáváme

$$g\left(\frac{x_i s_i}{\tau}\right) \leq \bar{K} - \sum_{j \neq i} g\left(\frac{x_j s_j}{\tau}\right) \leq \bar{K}.$$

Protože platí (a) a (c), musí existovat číslo M tak, že

$$\frac{1}{M} \leq x_i s_i \leq M, \qquad 1 \leq i \leq n. \tag{2.28}$$

Po sečtení je

$$x^T s \ = \sum_{i=1}^{n} x_i s_i \leq nM. \tag{2.29}$$

Podle lemmatu 2 a podle (2.29) existuje číslo M_u tak, že $x_i \in (0, M_u\rangle$ a $s_i \in (0, M_u\rangle$ pro všechna $i = 1, \ldots, n$. Užijeme-li navíc (2.28), pak $x_i \geq \frac{1}{M.s_i} \geq \frac{1}{M.M_u}$ pro každé i. Podobně pro s. (2.27) platí, položíme-li $M_l := \frac{1}{M.M_u}$.

Nyní už je vše připraveno pro to, abychom konečně mohli dokázat, že pro každé $\tau > 0$ nabývá funkce f_τ na množině $\mathcal{H}^0$ svého minima, že je toto minimum jediné a že může být užito ke konstrukci řešení (2.4) - (2.7).

Věta 12 *Předpokládejme, že $\mathcal{F}^0 \neq \emptyset$ a nechť τ je libovolné kladné číslo. Pak funkce f_τ nabývá na množině $\mathcal{H}^0$ lokálního minima a soustava (2.4)–(2.7) má řešení.*

Důkaz: Jak jsme již ukázali, množina $\mathcal{L}_K$ je částí kompaktní podmnožiny množiny $\mathcal{H}^0$ a $f_\tau(x,s) > K$ pro všechna $(x,s) \in \mathcal{H}^0 - \mathcal{L}_K$. Z toho vyplývá, že f_τ nabývá na $\mathcal{H}^0$ svého minima. Protože f_τ je ostře konvexní, je toto minimum jediné.

Nyní ukažme, že argument minima funkce f_τ tvoří složky x a s řešení soustavy (2.4)– (2.7). Uvažovaný bod řeší problém

$$\begin{aligned} &\min f_\tau(x,s) \\ &\text{na množině } \left\{(x,y,s) : A\ x = b, A^T y + s\ = c, (x,s) > 0\right\}. \end{aligned} \tag{2.30}$$

Aplikujeme-li KKT podmínky (1.16)–(1.20), získáme soustavu

$$\frac{\partial}{\partial x} f_\tau(x,s) = A^T v \quad \Rightarrow \quad \frac{s}{\tau} - X^{-1}e = A^T v, \tag{2.31}$$

$$\frac{\partial}{\partial y} f_\tau(x,s) = A\ w \quad \Rightarrow \quad 0 = -A\ w, \tag{2.32}$$

$$\frac{\partial}{\partial s} f_\tau(x,s) = w \quad \Rightarrow \quad \frac{x}{\tau} - S^{-1}e = w, \tag{2.33}$$

kde X, S, e je obvyklé značení pro $\mathrm{diag}(x_1,\ldots,x_n)$, $\mathrm{diag}(s_1,\ldots,s_n)$, $(1,\ldots,1)^T$ a v, w jsou příslušné vektory Lagrangeových multiplikátorů. Z (2.32) a (2.33) získáme

$$A\left(\frac{x}{\tau} - S^{-1}e\right) = 0$$

a dále z (2.31) a (2.33) dostaneme

$$\left(\frac{1}{\tau}Xe - S^{-1}e\right)^T \left(\frac{1}{\tau}Se - X^{-1}e\right) = 0,$$

z čehož vyplývá

$$\begin{aligned} 0 &= \left(\frac{1}{\tau}Xe - S^{-1}e\right)^T \left(X^{-\frac{1}{2}}S^{\frac{1}{2}}\right)\left(X^{\frac{1}{2}}S^{-\frac{1}{2}}\right)\left(\frac{1}{\tau}Se - X^{-1}e\right) \\ &= \left|\left|\frac{1}{\tau}\left(XS\right)^{\frac{1}{2}}e - \left(XS\right)^{-\frac{1}{2}}e\right|\right|^2. \end{aligned}$$

Tedy i

$$\frac{1}{\tau}\left(XS\right)^{\frac{1}{2}}e - \left(XS\right)^{-\frac{1}{2}}e = 0 \qquad \text{a tudíž} \qquad XSe = \tau e.$$

Ukázali jsme, že argument minima (x,s) funkce f_τ spolu s vektorem y z (2.30) splňuje podmínky (2.4)– (2.7). Důkaz je hotov. □

Kapitola 3

Metody path–following

V kapitole 2 jsme popsali centrální cestu $\mathcal{C}$ jako křivku obsahující body (x_τ, y_τ, s_τ), která (pro $\tau \to 0$) vede do množiny primárně–duálních řešení Ω. Body z množiny Ω splňují KKT podmínky (1.9)–(1.12), zatímco body, které náleží centrální cestě jsou definovány vztahy, jenž se od podmínek KKT liší v hodnotě součinů $x_i s_i$. Body z $\mathcal{C}$ řeší konkrétně soustavu

$$\begin{aligned} Ax &= b, && (3.1)\\ A^T y + s &= c, && (3.2)\\ (x, s) &\geq 0, && (3.3)\\ x_i s_i &= \tau, \quad i = 1, \ldots, n. && (3.4) \end{aligned}$$

V kapitole 2 jsme také ukázali, že je-li úloha lineárního programování přípustná, má tato soustava pro pevně zvolenou hodnotu $\tau > 0$ právě jedno řešení (x_τ, y_τ, s_τ), ačkoli KKT podmínky (tj. podmínky (3.1)–(3.4) pro $\tau = 0$) můžou mít i řešení vícenásobná.

Slovní spojení "Metody path–following", jenž by bylo případně možné doslova přeložit jako "Metody následující cestu", je odvozeno z následujícího faktu. Všechny iterace, které získáme těmito metodami, náleží jistému pevně zvolenému okolí centrální cesty $\mathcal{C}$ a pro $\tau \to 0$ ji "následují" do množiny řešení Ω. Okolí $\mathcal{C}$ přitom volíme tak, aby neobsahovalo body ležící příliš blízko hranice nezáporného orthantu $(x, s) \geq 0$. Současně s tím v každé iteraci snížíme hodnotu parametru μ a to tím způsobem, že provedeme jeden krok Newtonovy metody směrem k bodu $(x_\tau, y_\tau, s_\tau) \in \mathcal{C}$, přičemž $\tau := \sigma\mu$, kde $\sigma \in \langle 0, 1\rangle$ je centrující parametr zavedený v kapitole 2. Nová hodnota μ_{k+1} je pak menší nebo rovna původní hodnotě μ_k a iterace $(x^{k+1}, y^{k+1}, s^{k+1})$ je blíže přesnému řešení, splňujícímu KKT podmínky.

Algoritmy uvedené v této kapitole generují ostře přípustné iterace(x^k, y^k, s^k), které splňují podmínky (3.1)–(3.3). Protože užíváme Newtonovu metodu, podmínka (3.4) přesně splněna není. Součiny $x_i s_i$ obecně nejsou stejné pro všechna i a (x^k, y^k, s^k) se proto odchylují od centrální cesty $\mathcal{C}$. Míru této odchylky zjistíme srovnáním jednotlivých součinů s jejich průměrnou hodnotou $\mu = (1/n)\sum x_i s_i = (x^T s)/n$. V konkrétním případě můžeme užít např. váženou normu (a scaled norm), definovanou vztahem

$$\frac{1}{\mu}||XSe - \mu e|| = \frac{1}{\mu}|| \begin{bmatrix} x_1 s_1 \\ \vdots \\ x_n s_n \end{bmatrix} - \left(\frac{x^T s}{n}\right) e|| \tag{3.5}$$

$\| \cdot\cdot \|$ většinou znamená 2-normu nebo ∞-normu. V obou případech platí, je-li $(1/\mu)$ $\| XSe - \mu e \| < 1$, pak $x > 0$, $s > 0$. (Je-li i-tá složka vektoru x nebo vektoru s rovna nule, je $\| XSe - \mu e \| \geq |x_i s_i - \mu| = \mu$). Hodnota μ plní roli „míry očekávaného přiblížení“, zmíněné v kapitole 2. Uvědomme si totiž, že pro body ležící na centrální cestě je hodnota výrazu (3.5) rovna nule.

Užijeme-li v (3.5) 2-normu a omezíme-li odchylku tak, aby byla menší nebo rovna pevně zvolené hodnotě $\theta \in \langle 0, 1)$, získáme okolí $\mathcal{N}_2(\theta)$ definované vztahem

$$\mathcal{N}_2(\theta) := \{(x, y, s) \in \mathcal{F}^0; \ \| XSe - \mu e \|_2 \leq \theta\mu\} \tag{3.6}$$

Užijeme-li v (3.5) ∞-normu, získáme obdobně okolí $\mathcal{N}_\infty(\theta)$.

Můžeme zavést ještě jedno okolí,

$$\mathcal{N}_{-\infty}(\gamma) := \{(x, y, s) \in \mathcal{F}^0; x_i s_i \geq \gamma\mu \quad \text{pro každé} \quad i = 1, \ldots, n\}, \tag{3.7}$$

kde $\gamma \in (0, 1)$,

a to na základě následující poznámky. Uvědomme si, že se pomocí (3.6) snažíme, aby pro žádné i nebyl rozdíl $\mu - x_i s_i$ příliš velký (tj. aby součiny nebyly o mnoho menší, než je jejich průměrná hodnota). Tedy aby se žádná ze složek vektoru (x, s) nepřiblížila k hranici nezáporného orthantu $(x, s) \geq 0$ předčasně, což znamená pro takové k, pro něž ještě není μ_k dostatečně malé. Přesně totéž lze zajistit pomocí (3.7). Na rozdíl od $\mathcal{N}_2(\theta)$ resp. $\mathcal{N}_\infty(\theta)$ klade okolí $\mathcal{N}_{-\infty}(\gamma)$ na hodnoty $x_i s_i$ pouze jednostranné omezení. Někdy se proto také nazývá jednostranné ∞-norma okolí. Typické hodnoty parametrů θ a γ jsou $\theta = 0.5$, $\gamma = 10^{-3}$.

Jestliže bod leží v $\mathcal{N}_{-\infty}(\gamma)$, musí být každý ze součinů $x_i s_i$ alespoň malým násobkem jejich průměrné hodnoty μ. Tato podmínka ve skutečnosti není příliš omezující a pro γ blízké nule obsahuje $\mathcal{N}_{-\infty}(\gamma)$ většinu bodů přípustné oblasti $\mathcal{F}$. Okolí $\mathcal{N}_2(\theta)$ je restriktivnější. Některé body z $\mathcal{F}^0$ nepatří do $\mathcal{N}_2(\theta)$ pro žádné θ libovolně blízké 1. Tyto vlastnosti ještě zmíníme v dalším textu.

Metody path–following se drží základního PD schématu uvedeného v kapitole 2. V každé z nich je pevně zvoleno jedno z okolí $\mathcal{N}_2(\theta)$, $\mathcal{N}_\infty(\theta)$, $\mathcal{N}_{-\infty}(\gamma)$ a centrující parametr σ. Délka kroku α je pak volena tak, aby všechny iterace (x^k, y^k, s^k) byly prvky zvoleného okolí.

V této kapitole podrobněji popíšeme tři metody path–following. Jednak metodu path–following s krátkým krokem (Algoritmus SPF – Short–step path–following) a metodu path–following prediktor–korektor (Algoritmus PC), u obou volíme okolí $\mathcal{N}_2(\theta)$, a dále metodu path–following s dlouhým krokem (Algoritmus LPF - Long–step path–following), kde volíme okolí$\mathcal{N}_{-\infty}(\gamma)$.

Metoda SPF, nejjednodušší ze všech metod vnitřního bodu, volí pro všechna k konstantní hodnotu centrujícího parametru $\sigma_k = \sigma$ a konstantní hodnotu délky kroku $\alpha^k = 1$. (Zde uvedená analýza vychází z článku Monteiro & Adler [9] a je popsána v knize S.Wright [2].)

Metoda PC střídá dva typy kroků: prediktorové kroky, které zmenšují hodnotu μ, ale zvětšují hodnotu výrazu (3.5), což znamená zhoršení centrality aproximace, a korektorové kroky, které hodnotu parametru μ nemění, ale centralitu aproximace opět zlepšují. Algoritmem PC se zabývali mnozí autoři (např. Monteiro & Adler [9], Sonnevend & Stoer & Zhao [11, 12]), ale úplně poprvé byl analyzován v práci Mizuno & Todd & Ye [10]. Někdy je proto také nazýván Mizunův-Toddův-Yeův algoritmus prediktor–korektor. Důvodem je jeho odlišení od Mehrotrova algoritmu prediktor–korektor (viz např. S.Wright [2], D.F.Shano [14].)

Hlavní nevýhodou okolí $\mathcal{N}_2(\theta)$ je jeho restriktivní charakter. Z definice (3.6) vidíme, že pro každý bod (x, y, s), ležící v $\mathcal{N}_2(\theta)$ musí platit vztah

$$\sum_{i=1}^{n} \left[(x_i s_i/\mu) - 1\right]^2 \leq \theta^2 < 1,$$

který neznamená nic jiného, než že součet čtverců všech relativních odchylek $x_i s_i$ od jejich průměrné hodnoty nepřesáhne hodnotu 1. Ať je $\theta \in \langle 0, 1)$ jakkoli blízko jedničce, zahrnuje okolí $\mathcal{N}_2(\theta)$ pouze malou část množiny ostře přípustných řešení $\mathcal{F}^0$.

Oproti tomu okolí $\mathcal{N}_{-\infty}(\gamma)$ je podstatně expanzivnější a pro malé hodnoty γ zahrnuje téměř všechny body z množiny ostře přípustných řešení $\mathcal{F}^0$. Na výběru tohoto okolí je založena metoda LPF. Tato metoda volí hodnotu centralizujícího parametru σ_k menší než metoda SPF. Podmínka $x_i^{k+1} s_i^{k+1} = \sigma_k \mu_k$ je blíže KKT podmínce $x_i s_i = 0$ a metoda LPF je v jistém smyslu „agresivnější“ než metoda SPF. Směr $(\Delta x, \Delta y, \Delta s)$ je řešením soustavy (2.11) a délka kroku α^k je volena jako největší možná hodnota α, pro niž je $(x^k, y^k, s^k) + \alpha(\Delta x^k, \Delta y^k, \Delta s^k)$ prvkem $\mathcal{N}_{-\infty}(\gamma)$. Složitost LPF je ovšem $O\left(n \log(1/\epsilon)\right)$, zatímco složitost SPF resp. PC je $O\left(\sqrt{n} \log(1/\epsilon)\right)$.

Ačkoli k-tá iterace metod path–following směřuje k bodu $(x_\tau, y_\tau, s_\tau) \in \mathcal{C}$, pro který $x_i s_i = \tau_k = \sigma_k \mu_k$, jen zřídka se podaří tohoto bodu dosáhnout přesně. Důvodem je rozdíl mezi nelineární soustavou (3.1)–(3.4) a její lineární aproximací, která vede na soustavu (2.11). Tento rozdíl lze kvantitativně popsat pomocí součinů $\Delta x_i \Delta s_i$. V dalších odstavcích uvedeme některé odhady těchto součinů. Dále se zaměříme na konvergenci posloupnosti parametrů $\{\mu_k\}$ k nule a na konvergenci posloupnosti iterací $\{(x^k, y^k, s^k)\}$. Rovněž ukážeme, že složky (x^k, s^k) jsou omezené a hromadný bod posloupnosti $\{(x^k, s^k)\}$ je řešením úlohy lineárního programování, splňujícím podmínku komplementarity.

3.1 Metoda path–following s krátkým krokem (SPF)

Jako první uveďme algoritmus SPF. Metoda startuje z bodu $(x^0, y^0, s^0) \in \mathcal{N}_2(\theta)$ a užívá konstantní hodnoty $\alpha^k = 1$, $\sigma_k = \sigma$, kde θ a σ splňují určité vztahy popsané níže. Všechny iterace (x^k, y^k, s^k) jsou prvky $\mathcal{N}_2(\theta)$ a míra duality μ_k konverguje lineárně k nule s kvocientem $(1 - \sigma)$.

Metoda vychází ze základního PD schématu. Parametry θ a σ nabývají speciálních hodnot, jejichž volbu zdůvodníme později.

Algoritmus SPF

Dáno $\theta := 0.4$, $\sigma := 1 - 0.4/\sqrt{n}$, $(x^0, y^0, s^0) \in \mathcal{N}_2(\theta)$
for $k = 0, 1, 2, \ldots$
položme $\sigma_k := \sigma$ a řešme soustavu (2.11) pro $(\Delta x^k, \Delta y^k, \Delta s^k)$
definujme $(x^{k+1}, y^{k+1}, s^{k+1}) := (x^k, y^k, s^k) + (\Delta x^k, \Delta y^k, \Delta s^k)$
end(for).

Několik prvních iterací algoritmu SPF je znázorněno na obr. 3.1. Pro dvoudimenzionální problém ($n = 2$) tvoří osy soustavy souřadnic, poněkud neobvykle, součiny $s_1 x_1$, $x_2 s_2$. Centrální cesta je přímka, vycházející z počátku a svírající s osou $x_1 s_1$ úhel $\pi/4$. V této (nelineární) souřadné soustavě se vektor $(\Delta x^k, \Delta y^k, \Delta s^k)$ transformuje na obecně nelineární křivku. Přesné řešení úlohy lineárního programování je bod (0,0).

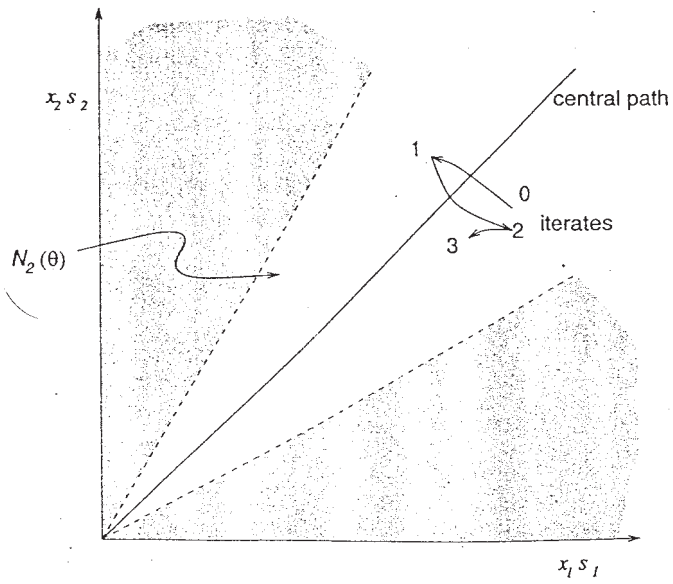

Figure 3.1: Iterates of Algorithm SPF, plotted in (xs) space.

Z obrázků 3.1, 3.2 a 3.3 je vidět, jak v jednotlivých algoritmech omezuje hranice okolí délku kroku.

Na tomto místě uvedeme několik lemmat a vět, které popisují vlastnosti algoritmu SPF. Jejich důkazy lze nalézt v knize S.Wright [2]. Největším problémem je dokázat, že všechny iterace algoritmu SPF jsou prvky okolí $\mathcal{N}_2(\theta)$. O této vlastnosti algoritmu hovoří lemmata 4 a 5 a věta 13. Z lemmatu 3 získáme lineární konvergenci posloupnosti parametrů $\{\mu_k\}$.

Zaveďme ještě následující označení:

$$(x(\alpha), y(\alpha), s(\alpha)) := (x, y, s) + \alpha(\Delta x, \Delta y, \Delta s), \tag{3.8}$$

$$\mu(\alpha) := (x(\alpha))^T (s(\alpha))/n. \tag{3.9}$$

Lemma 3 *Buď směr $(\Delta x, \Delta y, \Delta s)$ definován řešením soustavy (2.10). Pak je*

$$(\Delta x)^T \Delta s = 0 \tag{3.10}$$

a

$$\mu(\alpha) = (1 - \alpha\,(1 - \sigma))\,\mu. \tag{3.11}$$

Pro speciální volby parametrů $\sigma_k = 1 - 0.4/\sqrt{n}$ a $\alpha^k = 1$ algoritmu SPF dostáváme na základě (3.11) pro dvě po sobě následující hodnoty parametru μ vztah

$$\mu_{k+1} = \sigma_k \mu_k = (1 - 0.4/\sqrt{n})\mu_k \quad k = 0, 1, 2, \ldots \tag{3.12}$$

jehož důsledkem je globální lineární konvergence posloupnosti $\{\mu_k\}$ k nule.

Následující tvrzení se týkají vlastnosti $(x^k, y^k, s^k) \in \mathcal{N}_2(\theta)$ pro každé k. Z lemmatu 3 víme, že $\sum \Delta x_i \Delta s_i = 0$, což ovšem neznamená, že jsou nulové jednotlivé součiny $\Delta x_i \Delta s_i$. Horní odhad normy vektoru, jehož složky tvoří právě tyto součiny, uvádí následující lemma.

Lemma 4 *Je-li (x, y, s) prvkem $\mathcal{N}_2(\theta)$, pak*

$$||\Delta X \Delta S e|| \leq \frac{\theta^2 + n\,(1 - \sigma)^2}{2^{3/2}\,(1 - \theta)}\mu. \tag{3.13}$$

Důsledkem lemmatu 4 je lemma 5, které dává odhad vzdálenosti bodu $(x(\alpha), y(\alpha), s(\alpha))$ od centrální cesty.

Lemma 5 *Je-li (x, y, s) prvkem $\mathcal{N}_2(\theta)$, pak*

$$\begin{aligned} ||X(\alpha)S\ (\alpha)e - \mu(\alpha)e|| &\leq |1 - \alpha|||XSe - \mu e|| + \alpha^2||\Delta X \Delta S e|| && (3.14)\\ &\leq |1 - \alpha|\theta\mu + \alpha^2 \left[\frac{\theta^2 + n\,(1 - \sigma)^2}{2^{3/2}\,(1 - \theta)}\right]\mu. && (3.15) \end{aligned}$$

Věta 13 objasňuje vztah mezi θ a σ a ukazuje, že i při volbě délky kroku $\alpha = 1$ ve směru $(\Delta x, \Delta y, \Delta s)$ leží následující iterace v množině $\mathcal{N}_2(\theta)$.

Věta 13 *Zvolme parametry $\theta \in (0, 1)$ a $\sigma \in (0, 1)$ tak, aby splňovaly nerovnost*

$$\frac{\theta^2 + n\,(1 - \sigma)^2}{2^{3/2}\,(1 - \theta)} \leq \sigma\theta. \tag{3.16}$$

Pak platí: je-li $(x, y, s) \in \mathcal{N}_2(\theta)$, je $(x(\alpha), y(\alpha), s(\alpha)) \in \mathcal{N}_2(\theta)$ pro všechna $\alpha \in \langle 0, 1\rangle$.

Důkaz: Dosaďme (3.16) do (3.15). Pro $\alpha \in \langle 0,1\rangle$ získáme

$$\begin{aligned} ||X(\alpha)S\ (\alpha)e - \mu(\alpha)e|| &\leq (1-\alpha)\,\theta\mu + \alpha^2\sigma\theta\mu \\ &\leq (1-\alpha+\sigma\alpha)\,\theta\mu \quad (\text{protože } \alpha \in \langle 0,1\rangle) \\ &= \theta\mu(\alpha) \quad (\text{podle (3.11) }). \end{aligned} \tag{3.17}$$

Bod $(x(\alpha), y(\alpha), s(\alpha))$ tedy splňuje podmínku z definice okolí $\mathcal{N}_2(\theta)$.
Zbývá ještě dokázat, že $(x(\alpha), y(\alpha), s(\alpha)) \in \mathcal{F}^0$. Je snadné ověřit, že

$$Ax(\alpha) = b, \qquad A^T y(\alpha) + s(\alpha) = c.$$

Ověřme ještě vztah $(x(\alpha), s(\alpha)) > 0$. Nejprve však připomeňme, že $(x(0), s(0)) = (x, s) > 0$. Podle (3.17) je

$$x_i(\alpha)s_i(\alpha) \geq (1-\theta)\,\mu(\alpha) = (1-\theta)\,(1-\alpha\,(1-\sigma))\,\mu > 0. \tag{3.18}$$

Poslední nerovnost je důsledkem toho, že $\theta \in (0,1), \alpha \in (0,1\rangle$ a $\sigma \in (0,1)$. Z (3.18) vyplývá, že ani $x_i(\alpha)$, ani $s_i(\alpha)$ nemůže být rovné nule (pro žádné i) a tedy $(x(\alpha), s(\alpha)) > 0$ pro každé $\alpha \in \langle 0,1\rangle$. □

Teď zbývá už jenom ověřit podmínku (3.16) pro speciální hodnoty θ a σ z algoritmu SPF. Po dosazení $\theta = 0.4$ a $\sigma = 1 - 0.4/\sqrt{n}$ se snadno přesvědčíme, že (3.16) platí pro všechna $n \geq 1$.

3.2 Metoda prediktor–korektor (PC)

Algoritmus SPF volí parametry $\sigma_k := \sigma$, které leží ostře mezi nulou a jedničkou. Tato volba zaručí zlepšení centrality následující iterace a snížení hodnoty μ v jednom kroku. Algoritmus PC rozděluje tuto úlohu na dvě a pravidelně střídá dva typy kroků:
1. prediktor ($\sigma_k = 0$) pro snížení hodnoty μ
2. korektor ($\sigma_k = 1$) pro zlepšení centrality následující iterace.

Další významnou vlastností algoritmu PC je volba dvou okolí $\mathcal{N}_2(\theta)$, kde první z nich je obsažené uvnitř druhého. Sudé iterace (tj. body (x^k, y^k, s^k), k -sudé) jsou prvky vnitřního okolí, zatímco liché iterace jsou prvky vnějšího okolí.

Pro ilustraci rozeberme nyní podrobněji první dvě iterace algoritmu PC. Počáteční přiblížení (x^0, y^0, s^0) nechť je prvkem vnitřního okolí. Nejprve položme $\sigma_0 := 0$ a spočtěme $(\Delta x, \Delta y, \Delta s)$ pro prediktor. Ve směru tohoto vektoru postupujme tak dlouho, dokud nedosáhneme hranice vnějšího okolí. V tomto bodě pak definujme nové přiblížení (x^1, y^1, s^1). Nyní položme $\sigma_1 := 1$ a spočtěme $(\Delta x, \Delta y, \Delta s)$ pro korektor. V tomto směru zvolme jednotkový krok a následující iteraci pak definujme jako $(x^2, y^2, s^2) := (x^1, y^1, s^1) + \alpha(\Delta x, \Delta y, \Delta s)$, kde $\alpha := 1$. Bod (x^2, y^2, s^2) leží opět ve vnitřním okolí. Právě uvedený dvoukrokový cyklus se neustále opakuje a generuje posloupnos $\{(x^k, y^k, s^k)\}$, jejíž sudé prvky leží uvnitř vnitřního okolí a liché prvky na hranici vnějšího okolí.

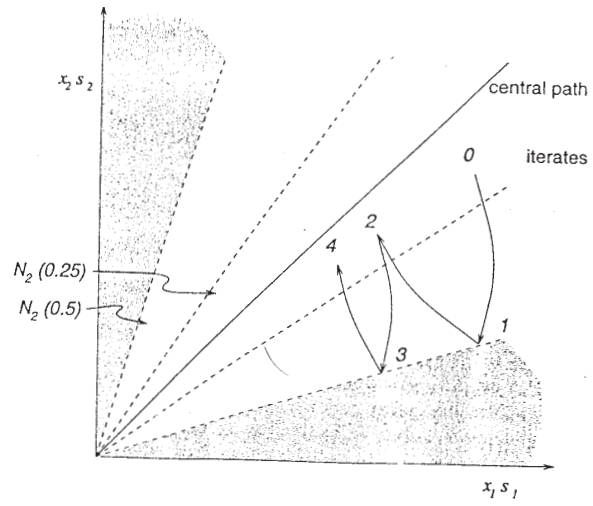

Figure 3.2: Iterates of Algorithm PC, plotted in (xs) space.

Několik prvních kroků cyklu je znázorněno na obr. 3.2. Přesné řešení úlohy leží v bodě (0,0).

Prediktorové kroky zmenšují hodnotu μ $(1-\alpha)$-krát, kde α je délka kroku. Korektorové kroky ponechají hodnotu μ nezměněnou, ale zaručí, že následující přiblížení leží opět ve vnitřním okolí. Tím poskytnou více „prostoru“ pro další (prediktorový) krok.

Formální popis algoritmu PC opět vychází ze základního PD schématu z kapitoly 2. Pro jednoduchost zvolme za vnitřní okolí $\mathcal{N}_2(0.25)$ a za vnější $\mathcal{N}_2(0.5)$. Hodnoty θ je možné volit i jinak, splňují-li příslušná okolí podmínky uvedené níže.

Algoritmus PC

Dáno $(x^0, y^0, s^0) \in \mathcal{N}_2(0.25)$
for $k = 0, 1, 2, \ldots$
if k sudé (* prediktor *)
položme $\sigma_k := 0$ a řešme soustavu (2.11) pro $(\Delta x^k, \Delta y^k, \Delta s^k)$;
zvolme α^k jako největší $\alpha \in \langle 0, 1\rangle$, pro nějž platí

$$(x^k(\alpha), y^k(\alpha), s^k(\alpha)) \in \mathcal{N}_2(0.5) \tag{3.19}$$

a definujme

$$(x^{k+1}, y^{k+1}, s^{k+1}) := (x^k(\alpha^k), y^k(\alpha^k), s^k(\alpha^k))$$

else (* korektor *)
položme $\sigma_k := 1$ a řešme soustavu (2.11) pro $(\Delta x^k, \Delta y^k, \Delta s^k)$
a definujme

$$(x^{k+1}, y^{k+1}, s^{k+1}) := (x^k, y^k, s^k) + (\Delta x^k, \Delta y^k, \Delta s^k)$$

end(if)
end(for).

Analýza metody PC je podobná analýze metody SPF. Uveďme tedy pouze některá doplňující lemmata. Důkazy lze opět nalézt v knize S.Wright [2].

Chování prediktorových kroků popisuje následující lemma. Uvádí dolní odhad pro délku kroku α a odhad hodnoty μ pro následující iteraci.

Lemma 6 *Předpokládejme, že (x, y, s) je prvkem $\mathcal{N}_2(0.25)$ a $(\Delta x, \Delta y, \Delta s)$ je směr definovaný soustavou (2.10) pro hodnotu $\sigma = 0$. Pak $(x(\alpha), y(\alpha), s(\alpha)) \in \mathcal{N}_2(0.5)$ pro všechna $\alpha \in \langle 0, \hat{\alpha}\rangle$, kde*

$$\tilde{\alpha} = \min\left(\frac{1}{2}, \left(\frac{\mu}{8||\Delta X \Delta S e||}\right)^{1/2}\right). \tag{3.20}$$

Délka kroku pro prediktor je tedy alespoň $\hat{\alpha}$ a nová hodnota $\mu(\alpha^k)$ je nejvýše $(1-\hat{\alpha})\mu$.

K nalezení dolní hranice pro $\hat{\alpha}$ můžeme využít lemmatu 4. Položme pro náš případ $\theta := 0.25$ a $\sigma := 0$. Získáme odhad

$$\frac{\mu}{8||\Delta X \Delta S e||} \geq \frac{2^{3/2}(1-0.25)}{8\left((0.25)^2 + n\right)} = \frac{3\sqrt{2}}{1+16n} \geq \frac{0.16}{n},$$

je-li $n \geq 1$. Potom podle (3.20)

$$\tilde{\alpha} \geq \min\left(\frac{1}{2}, \left(\frac{0.16}{n}\right)^{1/2}\right) = \frac{0.4}{\sqrt{n}}.$$

Poněvadž prediktorové kroky odpovídají sudým iteračním indexům, můžeme psát

$$\mu_{k+1} \leq (1 - 0.4/\sqrt{n})\mu_k \quad k = 0, 2, 4, \ldots \tag{3.21}$$

Lemma 7 popisuje chování korektorových kroků. Ukazuje, že korektor „přesouvá“ iterace z vnějšího okolí do vnitřního, aniž by změnil hodnotu μ.

Lemma 7 *Předpokládejme, že (x, y, s) je prvkem $\mathcal{N}_2(0.5)$ a $(\Delta x, \Delta y, \Delta s)$ je směr definovaný soustavou (2.10) pro hodnotu $\sigma = 1$. Pak*

$$(x(1), y(1), s(1)) \in \mathcal{N}_2(0.25) \quad \text{a} \quad \mu(1) = \mu.$$

Lze dokázat (důkaz opět viz S.Wright [2]), že složitost algoritmu PC je stejná jako složitost algoritmu SPF. Algoritmus PC je ale určitým zlepšením algoritmu SPF a to z důvodu větší adaptibility délky prediktorového kroku. Zatímco u SPF jsou hodnoty α^k za všech okolností stejné, u PC se mění v závislosti na prediktorovém směru $(\Delta x, \Delta y, \Delta s)$. Hodnotu α můžeme volit větší pro takový směr, v němž lze více snížit hodnotu μ, aniž by následujícíiterace ležela mimo přípustné okolí. Pro vzrůstající k se takové směry vyskytují stále častěji a hodnotu α lze pak volit velmi blízko 1. Posloupnost parametrů $\{\mu_k\}$ konverguje k nule superlineárně.

I přes tuto vlastnost je algoritmus PC stále ještě omezující, a sice z důvodu volby okolí $\mathcal{N}_2$. Uveďme proto jestě jeden algoritmus, který také umožňuje adaptivně volit hodnotu alfa, ale navíc používá méně omezující okolí $\mathcal{N}_{-\infty}(\gamma)$.

3.3 Metoda path–following s dlouhým krokem (LPF)

Algoritmus LPF generuje posloupnost iterací v okolí $\mathcal{N}_{-\infty}(\gamma)$, které pro γ dost malá (řekněme 10^{-3}) obsahuje téměř všechny body množiny ostře přípustných řešení $\mathcal{F}^0$. V každé iteraci volí hodnotu centralizujícího parametru σ_k tak, aby ležela mezi dvěma pevně stanovenými hodnotami $0 < \sigma_{\min} < \sigma_{\max} < 1$. Výběr směru získáme, jako obvykle, vyřešením soustavy (2.11). Hodnotu α^k volíme jako největší možnou délku kroku, pro niž leží následující iterace v $\mathcal{N}_{-\infty}(\gamma)$. Formální tvar algoritmu je následující:

Algoritmus LPF

Dáno γ, $\sigma_{\min}$, $\sigma_{\max}$, kde $\gamma \in (0,1)$, $0 < \sigma_{\min} < \sigma_{\max} < 1$ a $(x^0, y^0, s^0) \in \mathcal{N}_{-\infty}(\gamma)$
for $k = 0, 1, 2, \ldots$
zvolme $\sigma_k \in \langle \sigma_{\min}, \sigma_{\max} \rangle$ a řešme soustavu (2.11) pro $(\Delta x^k, \Delta y^k, \Delta s^k)$
zvolme α^k jako největší $\alpha \in \langle 0, 1 \rangle$, pro nějž platí

$$(x^k(\alpha), y^k(\alpha), s^k(\alpha)) \in \mathcal{N}_{-\infty}(\gamma) \tag{3.22}$$

definujme

$$(x^{k+1}, y^{k+1}, s^{k+1}) := (x^k(\alpha^k), y^k(\alpha^k), s^k(\alpha^k))$$

end(for).

Několik prvních kroků je opět znázorněno na následujícím obrázku.

Jak ukazuje obrázek (a jak potvrdí analýza) dolní hranice $\sigma_{\min}$ zaručuje toto: každou iteraci startujme z bodu na hranici a ve směru vektoru$(\Delta x^k, \Delta y^k, \Delta s^k)$ se pohybujeme nejprve dovnitř oblasti. To znamená, že zvolíme-li malou délku kroku α, zlepšíme centralitu iterace. Pro příliš velké hodnoty α se dostaneme znovu mimo okolí, poněvadž chyba, které se dopustíme aproximací nelineárního sytému (3.1)–(3.4) lineární soustavou (2.10), je pro rostoucí α výraznější. Nicméně přesto jsme pro každou iteraci schopni zaručit (v daném směru) jistý minimální krok α_{min}, pro nějž $(x(\alpha_{min}), y(\alpha_{min}), s(\alpha_{min}))$ ještě neleží na hranici okolí $\mathcal{N}_{-\infty}(\gamma)$.

V lemmatu 8 a větě 14 nalezneme dolní hranici pro toto α^k a odhad pro snížení hodnoty μ v každé z iterací.

Uvědomme si, že okolí $\mathcal{N}_2(\theta)$ a $\mathcal{N}_{-\infty}(\gamma)$ jsou na obrázcích 3.1, 3.2 a 3.3 reprezentovány stejně. Obě jsou ohraničena přímkami, vycházejícími z počátku. Tato podobnost ovšem platí pouze pro $n = 2$. Pro větší n mohou mít okolí zcela odlišné tvary.

Lemma 8 *Je-li* (x, y, s) *prvkem* $\mathcal{N}_{-\infty}(\gamma)$, *pak*

$$||\Delta X \Delta S e|| \leq 2^{-3/2} \left(1 + \frac{1}{\gamma}\right) n\mu. \tag{3.23}$$

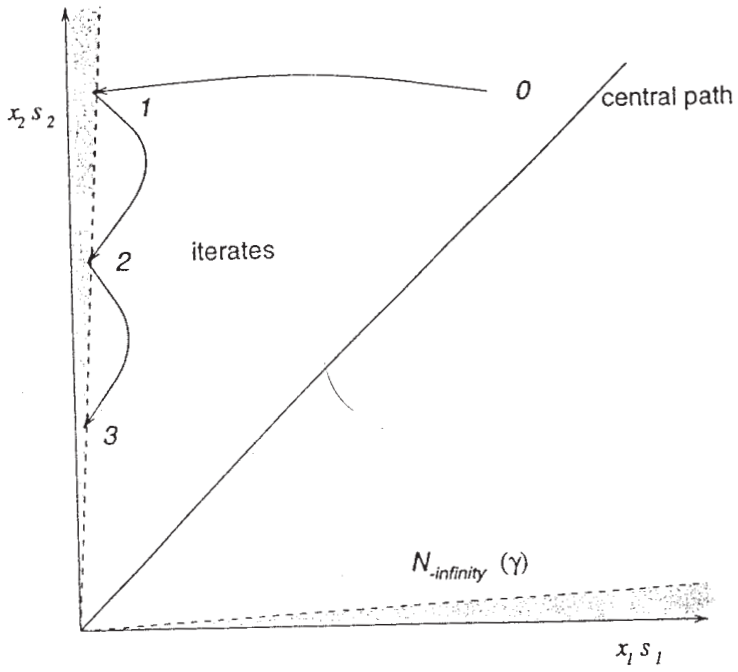

Figure 3.3: Iterates of Algorithm LPF, plotted in (xs) space.

Srovnejme tento odhad s odhadem uvedeným v Lemmatu 4. Zjistíme, že omezení (3.13) je těsnější, což je způsobeno větší restriktivností okolí $\mathcal{N}_2(\theta)$. Rozdíl v omezeních (3.13) a (3.23) je důvodem pro rozdílné složitosti algoritmů SPF a LPF.

Nadešel čas ukázat, že $\left(x^k(\alpha), y^k(\alpha), s^k(\alpha)\right) \in \mathcal{N}_{-\infty}(\gamma)$ pro všechna

$$\alpha \in \left\langle 0,\ 2^{3/2}\gamma\frac{1-\gamma}{1+\gamma}\frac{\sigma_k}{n} \right\rangle \tag{3.24}$$

a tudíž lze v každém kroku volit

$$\alpha_k \leq 2^{3/2}\frac{\sigma_k}{n}\gamma\frac{1-\gamma}{1+\gamma}. \tag{3.25}$$

Pro každé $i = 1, 2, \ldots, n$ plyne z lemmatu 3.13, že

$$|\Delta x_i^k \Delta s_i^k| \leq ||\Delta X^k \Delta S^k e||_2 \leq 2^{-\frac{3}{2}}\left(1+\frac{1}{\gamma}\right) n\mu_k. \tag{3.26}$$

Užijeme-li vztahu

$$S\ \Delta x + X\Delta s\ = -XSe + \sigma\mu e,$$

získáme z vlastnosti $x_i^k s_i^k \geq \gamma\mu_k$ a z (3.26)

$$\begin{aligned} x_i^k(\alpha)s_i^k(\alpha) &= \left(x_i^k + \alpha\Delta x_i^k\right)\left(s_i^k + \alpha\Delta s_i^k\right) \\ &= x_i^k s_i^k + \alpha\left(x_i^k\Delta s_i^k + s_i^k\Delta x_i^k\right) + \alpha^2\Delta x_i^k\Delta s_i^k \\ &\geq x_i^k s_i^k\left(1-\alpha\right) + \alpha\sigma_k\mu_k - \alpha^2|\Delta x_i^k\Delta s_i^k| \\ &\geq \gamma\left(1-\alpha\right)\mu_k + \alpha\sigma_k\mu_k - \alpha^2 2^{-\frac{3}{2}}\left(1+\frac{1}{\gamma}\right)n\mu_k. \end{aligned}$$

Navíc, podle (3.11), je

$$\mu_k(\alpha) = \left(1-\alpha\left(1-\sigma_k\right)\right)\mu_k.$$

Výsledně tedy podmínka z definice $\mathcal{N}_{-\infty}(\gamma)$

$$x_i^k(\alpha)s_i^k(\alpha) \geq \gamma\mu_k(\alpha) \tag{3.27}$$

platí, předpokládáme-li, že

$$\gamma\left(1-\alpha\right)\mu_k + \alpha\sigma_k\mu_k - \alpha^2 2^{-\frac{3}{2}}\left(1+\frac{1}{\gamma}\right)n\mu_k \geq \gamma\left(1-\alpha+\alpha\sigma_k\right)\mu_k,$$

čili, po úpravě,

$$\alpha\sigma_k\mu_k\left(1-\gamma\right) \geq \alpha^2 2^{-3/2}n\mu_k\left(1+\frac{1}{\gamma}\right),$$

což platí, je-li

$$\alpha \leq \frac{2^{3/2}}{n}\sigma_k\gamma\frac{1-\gamma}{1+\gamma}.$$

Tím jsme dokázali, že $\left(x^k(\alpha), y^k(\alpha), s^k(\alpha)\right)$ splňuje podmínku (3.27) z definice $\mathcal{N}_{-\infty}(\gamma)$, leží-li α v intervalu určeném (3.24). Podobně, jako v důkazu věty 3.10, můžeme ověřit, že pro tato α leží bod $\left(x^k(\alpha), y^k(\alpha), s^k(\alpha)\right)$ v množině $\mathcal{F}^0$. Tím jsme dokázali (3.24) a (3.25).

Věta 14 *Nechť jsou dány parametry γ, $\sigma_{\min}$, $\sigma_{\max}$ z algoritmu LPF. Pak existuje konstanta δ nezávislá na n tak, že*

$$\mu_{k+1} \leq (1-\delta/n)\mu_k$$

pro všechna $k \geq 0$.

Důkaz: Využijme toho, co už je dokázáno. Z (3.11) a (3.25) získáme

$$\begin{aligned} \mu_{k+1} &= \left(1-\alpha_k\left(1-\sigma_k\right)\right)\mu_k \\ &\leq \left(1-\frac{2^{3/2}}{n}\gamma\frac{1-\gamma}{1+\gamma}\sigma_k\left(1-\sigma_k\right)\right)\mu_k. \end{aligned} \tag{3.28}$$

Funkce $\sigma(1-\sigma)$ je konkávní kvadratická funkce v σ a tedy na každém intervalu nabývá minima v jednom z krajních bodů. Z toho plyne

$$\sigma_k(1-\sigma_k) \geq \min\{\sigma_{min}(1-\sigma_{min}), \sigma_{max}(1-\sigma_{max})\}$$

pro všechna $\sigma_k \in \langle \sigma_{min}, \sigma_{max} \rangle$. Nyní stačí dosadit tento odhad do (3.28) a položit

$$\delta := 2^{3/2}\gamma\frac{1-\gamma}{1+\gamma}\min\{\sigma_{min}(1-\sigma_{min}), \sigma_{max}(1-\sigma_{max})\}.\square$$

3.4 Hromadné body posloupnosti iterací

Předchozí závěry, týkající se konvergence, se soustředily hlavně na konvergenci μ_k k nule. Zaměřme se nyní na posloupnost iterací $\{(x^k, y^k, s^k)\}$; její chování je komplikovanější, než by se na první pohled mohlo zdát. Hlavním úkolem bude ukázat, že posloupnost složek $\{(x^k, s^k)\}$ má hromadný bod. Platí-li toto, pak můžeme primárně–duální řešení zkonstruovat následujícím způsobem:
Nechť $\mathcal{K}$ je vybraná posloupnost taková, že $\lim_{k\in\mathcal{K}}(x^k, s^k) = (x^*, s^*)$. Pro každé $k \in \mathcal{K}$ platí

$$Ax^k = b, \quad c - s^k \in \text{Range}(A^T), \quad (x^k, s^k) > 0.$$

Přejděme k limitě a užijme faktu, že množina $\text{Range}(A^T)$ je uzavřená a že $\mu_k \to 0$. Pro (x^*, s^*) získáme

$$A\ x^* = b, \quad c - s^* \in \text{Range}(A^T), \quad (x^*, s^*) \geq 0, \quad (x^*)^T s^* = 0.$$

Tedy $c - s^* = A^T y^*$ pro nějaké y^*. Tyto vztahy odpovídají KKT podmínkám (1.9)–(1.12); $(x^*, y^*, s^*) \in \Omega$ je nalezeno.

V této části se podrobněji podívejme na limitní chování posloupnosti $\{(x^k, s^k)\}$, generované algoritmy SPF, PC a LPF. Ukážeme, že tato posloupnost je omezená a tedy má alespoň jeden hromadný bod. Navíc všechny hromadné body odpovídají ostře komplementárnímu řešení (x^*, y^*, s^*), tj. řešení, pro nějž

$$x_i^* > 0 \ \ (i \in \mathcal{B}) \quad s_i^* > 0 \ \ (i \in \mathcal{N}) \tag{3.29}$$

kde $\mathcal{B} \cup \mathcal{N}$ je rozklad množiny indexů $\{1, \ldots, n\}$ definovaný v (1.13), (1.14)

Lemma 9 *Buď $\mu_0 > 0$ a $\gamma \in (0,1)$. Potom pro všechny body (x, y, s) takové, že*

$$(x, y, s) \in \mathcal{N}_{-\infty}(\gamma) \subset \mathcal{F}^0, \quad \mu \leq \mu_0, \tag{3.30}$$

(kde $\mu = \frac{x^T s}{n}$), existují konstanty C_0 a C_1 tak, že

$$||(x, s)|| \leq C_0 \tag{3.31}$$

$$0 < x_i \leq \frac{\mu}{C_1} \quad (i \in \mathcal{N}), \qquad 0 < s_i \leq \frac{\mu}{C_1} \quad (i \in \mathcal{B}), \tag{3.32}$$

$$s_i \geq C_1\gamma \quad (i \in \mathcal{N}), \qquad x_i \geq C_1\gamma \quad (i \in \mathcal{B}). \tag{3.33}$$

Věta 15 *Nechť $\{(x^k, y^k, s^k)\}$ je posloupnost iterací, generovaná algoritmy SPF, PC nebo LPF a nechť posloupnost μ_k konverguje k nule pro $k \to \infty$. Pak je posloupnost složek $\{(x^k, s^k)\}$ omezená a má tedy alespoň jeden hromadný bod. Každý z hromadných bodů odpovídá ostře komplementárnímu primárně–duálnímu řešení úlohy lineárního programování.*

Důkaz: Iterace generované každým z algoritmů SPF, PC a LPF, leží v okolí $\mathcal{N}_{-\infty}(\gamma)$ pro vhodné γ. Pro algoritmus SPF je

$$(x^k, y^k, s^k) \in \mathcal{N}_2(0.4) \subset \mathcal{N}_{-\infty}(0.4).$$

Pro algoritmus PC leží všechny iterace v $\mathcal{N}_2(0.5)$, jenž je podmnožinou $\mathcal{N}_{-\infty}(0.5)$, algoritmus LPF vybírá přímo okolí $\mathcal{N}_{-\infty}(\gamma)$ pro $\gamma \in (0, 1)$. Pro každou z metod je navíc posloupnost μ_k nerostoucí, $\mu_k \leq \mu_0$ pro každé K. Každá z iterací (x^k, y^k, s^k) tudíž splňuje předpoklady lemmatu 9.

Omezenost posloupnosti $\left\{(x^k, s^k)\right\}$ plyne z (3.31). Je-li $(x^*, s^*) \in R^n \times R^n$ její hromadný bod, můžeme nalézt $y^* \in R^m$ tak, že $(x^*, y^*, s^*) \in \Omega$. Protože platí (3.33), musí být

$$s_i^* \geq C_1\gamma > 0 \quad (i \in \mathcal{N}), \qquad x_i^* \geq C_1\gamma > 0 \quad (i \in \mathcal{B}),$$

z čehož vyplývá striktní komplementarita řešení. □

Má-li úloha jednoznačné primárně–duální řešení, pak posloupnost iterací, vytvořená libovolnou ze tří uvedených metod, konverguje k tomuto řešení, jak je zřejmé z věty 15.

Kapitola 4

Nepřípustné metody vnitřního bodu

Až dosud jsme uvažovali pouze takové metody, pro které bylo počáteční přiblížení (x^0, y^0, s^0) ostře přípustné; speciálně splňovalo vztahy $Ax^0 = b$, $A^T y^0 + s^0 = c$. V důsledku nulové pravé strany v (2.10) pak tyto vztahy splňovaly i všechny další iterace. Pro většinu úloh je však obtížné ostře přípustné počáteční přiblížení najít. Existují úlohy lineárního programování, pro které takové body vůbec neexistují. Jako příklad můžeme uvést úlohu

$$\min(2x_1 + x_2) \quad \text{na množině} \quad \{x : x_1 + x_2 + x_3 = 5 \ \ \& \ \ x_1 + x_3 = 5 \ \ \& \ \ x \geq 0\}.$$

Snadno ze lze přesvědčit, že množina $\mathcal{F}^0$ je pro tuto úlohu prázdná. Všeobecně lze říci, že úlohy lineárního programování získané transformací obecné úlohy do standardního tvaru obvykle nemají ostře přípustná řešení.

Jeden ze způsobů, jak se uvedeným potížím vyhnout, popíšeme v této kapitole. Vytvoříme algoritmus, který nepožaduje, aby počáteční přiblížení (x^0, y^0, s^0) leželo v množině $\mathcal{F}^0$, ale pouze, aby platilo $(x^0, s^0) > 0$. Protože takové počáteční přiblížení leží obvykle mimo množinu přípustných řešení, mají všechny metody s touto vlastností přívlastek „nepřípustné“. V každém kroku pak vybíráme směr $(\Delta x, \Delta y, \Delta s)$ tak, abychom jednak zlepšili centralitu iterace, ale navíc aby následující iterace byla blíže k přípustné oblasti. Definujme nyní rezidua r_b a r_c následovně

$$r_b := Ax - b, \qquad r_c := A^T y + s \ - c \tag{4.1}$$

Rovnice pro směr $(\Delta x, \Delta y, \Delta s)$ pak mají tvar

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} -r_b \\ -r_c \\ -XSe + \sigma\mu e \end{bmatrix}. \tag{4.2}$$

Tímto opět provedeme jeden krok Newtonovy metody, směřující k bodu $(x_{\sigma\mu}, y_{\sigma\mu}, s_{\sigma\mu}) \in \mathcal{C}$. Jeho aproximací je bod

$$(\hat{x}, \hat{y}, \hat{s}) := (x, y, s) + \alpha(\Delta x, \Delta y, \Delta s)$$

Je-li možné volit délku kroku $\alpha = 1$, pak je bod $(\hat{x}, \hat{y}, \hat{s})$ již přípustný a stejně tak jsou přípustné i všechny další iterace.

Až do této doby jsme stále mluvili poměrně obecně. Ve zbytku kapitoly popíšeme pro názornost jednu konkrétní metodu - metodu IPF, která je „nepřípustnou“ verzí metody path–following s dlouhým krokem (LPF), uvedené v kapitole 3. Tato metoda je nejblíže metodám užívaným v praxi. Obecně leží všechny iterace (x^k, y^k, s^k), generované algoritmem IPF mimo přípustnou oblast, ačkoli jejich limita je, samozřejmě, přípustné (a optimální) řešení úlohy lineárního programování. Metoda konverguje dokonce i v případě, kdy je množina $\mathcal{F}^0$ prázdná.

Jak už jsem předeslala, jednotlivé kroky algoritmu získáme vyřešením modifikované Newtonovy soustavy (4.2) pro hodnoty parametru $\sigma > \varepsilon > 0$. Stejně jako u metody LPF chceme i nyní, aby všechny iterace ležely v určitém okolí centrální cesty. V našem případě vznikne toto okolí rozšířením okolí $\mathcal{N}_{-\infty}(\gamma)$, které užívala metoda LPF, o nepřípustné body. Délku kroku alfa omezíme dvěma novými podmínkami. Jednak budeme požadovat, aby chyba výrazů $Ax = b$ a $A^T y + s = c$ klesala k nule alespoň tak rychle jako hodnota míry duality μ. Za druhé pak zavedeme Armijovu podmínku o minimálním poklesu hodnoty μ v každé iteraci.

Analýza uvedená v následujících odstavcích ukáže, že posloupnost parametrů μ_k konverguje monotonně k nule pro libovolné počáteční přiblížení, pro nějž je $(x^0, s^0) > 0$. Zvolíme-li navíc $(x^0, y^0, s^0) = \rho(e, 0, e)$, kde ρ je dostatečně velké, dostaneme polynomiální složitost algoritmu. (Bod, pro nějž je $\mu_k \leq \epsilon$ získáme v $O(n^2 |\log \epsilon|)$ iteracích.) Algoritmus IPF není o mnoho komplikovanější než algoritmus LPF, jeho analýza ovšem vyžaduje příliš technických důkazů. Většinu z nich zde neuvádím; lze je nalézt např. v knize S.Wright [2].

4.1 Metoda IPF

Nejprve zaveďme rezidua předpisem (4.1). Počítáme-li tyto hodnoty pro bod (x^k, y^k, s^k), budeme je značit r_b^k resp. r_c^k. Jak už jsme jednou uvedli, algoritmus IPF užívá okolí $\mathcal{N}_{-\infty}(\gamma, \beta)$, které je rozšířením okolí $\mathcal{N}_{-\infty}(\gamma)$; definujme ho následovně

$$\mathcal{N}_{-\infty}(\gamma, \beta) := \{(x, y, s) : \quad ||(r_b, r_c)|| \leq \frac{||(r_b^0, r_c^0)||}{\mu_0} \beta\mu, \tag{4.3}$$

$$x_i s_i \geq \gamma\mu \quad , \quad 1 \leq i \leq n, \tag{4.4}$$

$$(x, s) > 0\} \tag{4.5}$$

kde $\gamma \in (0, 1)$ a $\beta \geq 1$ jsou dané parametry a (r_b^0, r_c^0) resp. μ_0 jsou hodnoty reziduí resp. míry duality pro počáteční přiblížení. Vztah (4.3) přepišme do tvaru

$$\frac{||(r_b, r_c)||}{||(r_b^0, r_c^0)||} \leq \frac{\mu}{\mu_0} \beta$$

Z toho je zřejmé, že aby v okolí $\mathcal{N}_{-\infty}(\gamma, \beta)$ ležel i bod (x^0, y^0, s^0), musí skutečně platit $\beta \geq 1$. Z výrazu (4.3) navíc vyplývá, že míra nepřípustnosti každého bodu z množiny $\mathcal{N}_{-\infty}(\gamma, \beta)$, vyjádřená normou vektoru reziduí $\| \ (r_b, r_c) \ \|$, je omezená

nějakým násobkem parametru μ. Zaručíme-li tedy, že všechny iterace (x^k, y^k, s^k) leží v množině $\mathcal{N}_{-\infty}(\gamma, \beta)$ a že posloupnost $\{\mu_k\}$ konverguje k nule pro $k \to \infty$, pak zřejmě pro $k \to \infty$ rovněž $r_b^k \to 0$ a $r_c^k \to 0$. Podmínku (4.4) zavádíme ze stejného důvodu jako u algoritmu LPF. Předchází tomu, aby součiny $x_i s_i$ byly (pro nízké hodnoty k) příliš blízko nule a zabraňuje tak selhání Newtonovy metody.

Algoritmus IPF

Dáno γ, β, $\sigma_{\min}$, $\sigma_{\max}$, kde $\gamma \in (0,1)$, $\beta \geq 1$, $0 < \sigma_{\min} < \sigma_{\max} < 0.5$ a (x^0, y^0, s^0), pro nějž $(x^0, s^0) > 0$.
for $k = 0, 1, 2, \ldots$
zvolme $\sigma_k \in \langle \sigma_{\min}, \sigma_{\max} \rangle$ a řešme soustavu

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} -r_b \\ -r_c \\ -XSe + \sigma\mu e \end{bmatrix} \tag{4.6}$$

zvolme α^k jako největší $\alpha \in \langle 0, 1 \rangle$, pro nějž

$$(x^k(\alpha), y^k(\alpha), s^k(\alpha)) \in \mathcal{N}_{-\infty}(\gamma, \beta) \tag{4.7}$$

a tak, aby platila Armijova podmínka

$$\mu_k(\alpha) \leq (1 - 0.01\alpha)\mu_k \tag{4.8}$$

definujme

$$(x^{k+1}, y^{k+1}, s^{k+1}) := (x^k(\alpha^k), y^k(\alpha^k), s^k(\alpha^k))$$

end(for).

Užíváme označení zavedené v (3.8), (3.9)

$$(x(\alpha), y(\alpha), s(\alpha)) := (x, y, s) + \alpha(\Delta x, \Delta y, \Delta s), \tag{4.9}$$

$$\mu(\alpha) := (x(\alpha))^T (s(\alpha))/n \tag{4.10}$$

Ve volbě délky kroku alfa existuje určitá volnost. Místo největší hodnoty, pro niž platí (4.7) a (4.8) můžeme volit hodnotu menší; tato volba může být v některých případech i výhodnější. Lze například volit $\alpha^k := \arg\min \mu(\alpha)$.

Než přikročíme k dalšímu, zaveďme ještě jedno užitečné označení.

$$\nu_0 := 1, \tag{4.11}$$

$$\nu_k := \prod_{j=0}^{k-1} (1 - \alpha_j)\,. \tag{4.12}$$

První dvě komponenty funkce $\mathcal{F}$, definované v (2.1), (2.2) jsou lineární, a proto je

$$r_b^k = Ax^k - b = A(x^{k-1} + \alpha^{k-1}\Delta x^{k-1}) - b = Ax^{k-1} - b + \alpha^{k-1}A\ \Delta x^{k-1}$$

Ze soustavy (4.6) dosaďme za výraz $A\Delta x^{k-1}$ hodnotu $-r_b^{k-1}$. Získáme $r_b^k = (1 - \alpha^{k-1})r_b^{k-1}$. Obdobně pro r_c^k. Výsledně tedy můžeme psát

$$\begin{aligned}(r_b^k, r_c^k) &= (1-\alpha^{k-1})(r_b^{k-1}, r_c^{k-1}) \\ &= (1-\alpha^{k-1})(1-\alpha^{k-2})(r_b^{k-2}, r_c^{k-2}) \\ &= \ldots\cdots = \\ &= \nu_k(r_b^0, r_c^0). \end{aligned} \tag{4.13}$$

Protože $(x^k, y^k, s^k) \in \mathcal{N}_{-\infty}(\gamma, \beta)$, je podle (4.13)

$$\frac{\nu_k||(r_b^0, r_c^0)||}{\mu_k} = \frac{||(r_b^k, r_c^k)||}{\mu_k} \leq \frac{||(r_b^0, r_c^0)||}{\mu_0}\beta.$$

Předpokládáme-li, že $(r_b^0, r_c^0) \neq 0$, platí nerovnost

$$\nu_k \leq \beta\frac{\mu_k}{\mu_0}. \tag{4.14}$$

Je-li $(r_b^0, r_c^0) = 0$, je počáteční přiblížení přípustné a stejně tak jsou přípustné i všechny iterace. Metoda IPF se zredukuje na metodu LPF z kapitoly 3. Pro jednoduchost uvažujme v dalším pouze případ $(r_b^0, r_c^0) \neq 0$.

4.2 Konvergence algoritmu IPF

Dokážeme-li, že existuje konstanta $\tilde{\alpha}$ tak, že $\alpha^k \geq \tilde{\alpha}$ pro všechna k, pak z podmínky (4.8) vyplývá

$$\mu_k(\alpha) \leq (1 - 0.01\alpha)\mu_k \leq (1 - 0.01\tilde{\alpha})\mu_k \quad \text{pro všechna } k \tag{4.15}$$

a tedy posloupnost $\{\mu_k\}$ konverguje Q-lineárně k nule. Ze vztahu (4.13) pak získáme

$$\| (r_b^k, r_c^k) \| \leq (1 - \tilde{\alpha}) \| (r_b^{k-1}, r_c^{k-1)} \| \tag{4.16}$$

a tedy posloupnost norem reziduí také konverguje k nule. Polynomiální složitost algoritmu lze dokázat na základě toho, že dolní hranice délky kroku $\tilde{\alpha}$ je inverzní polynomiální funkcí n, zvolíme-li počáteční přiblížení

$$(x^0, y^0, s^0) = \rho(e, 0, e), \tag{4.17}$$

kde ρ je takové, že

$$\| (x^*, s^*) \|_\infty \leq \rho \tag{4.18}$$

pro nějaké primárně–duální řešení (x^*, y^*, s^*). Většinou normu $\| (x^*, s^*) \|_\infty$ pochopitelně neznáme, vztahy (4.17) a (4.18) jsou přesto v praxi užitečné. Zvolíme-li počáteční přiblížení pro nějž je hodnota výrazu

$$\| (r_b^0, r_c^0) \| / \mu_0 \tag{4.19}$$

malá, získáme rychlejší konvergenci algoritmu. Pro vektory tvaru (4.17) je tento poměr řádově asi $1/\rho$.

Dříve, než uvedeme jedno pomocné lemma (důkaz viz S.Wright [2]), definujme pozitivně definintní diagonální matici D předpisem

$$D := (X)^{1/2}(S)^{-1/2}, \tag{4.20}$$

kde matice X a S mají stejný význam jako v předchozím textu. Budeme-li matici D vytvářet pro hodnoty X^k a S^k, budeme ji značit D^k.

Lemma 10 *Existuje kladná konstanta C_1 tak, že*

$$\| (D^k)^{-1}\Delta x^k \| \leq C_1 \mu_k^{1/2}, \quad \| D^k \Delta s^k \| \leq C_1 \mu_k^{1/2}, \tag{4.21}$$

pro každé k.

Nejdůležitějším tvrzením této kapitoly je bezpochyby lemma 11. Uveďme ho nyní a včetně důkazu.

Lemma 11 *Existuje konstanta $\tilde{\alpha} \in (0,1)$ tak, že pro každé $\alpha \in \langle 0, \tilde{\alpha} \rangle$ a pro každé $k \geq 0$ jsou splněny následující podmínky*

$$\left(x^k + \alpha\Delta x^k\right)^T \left(s^k + \alpha\Delta s^k\right) \geq (1-\alpha)\,(x^k)^T s^k, \tag{4.22}$$

$$\left(x_i^k + \alpha\Delta x_i^k\right)\left(s_i^k + \alpha\Delta s_i^k\right) \geq \frac{\gamma}{n}\left(x^k + \alpha\Delta x^k\right)^T \left(s^k + \alpha\Delta s^k\right), \tag{4.23}$$

$$\left(x^k + \alpha\Delta x^k\right)^T \left(s^k + \alpha\Delta s^k\right) \leq (1-0.01\alpha)\,(x^k)^T s^k. \tag{4.24}$$

Podmínky (4.7) a (4.8) jsou tedy splněny pro všechna $\alpha \in \langle 0, \tilde{\alpha} \rangle$ a pro všechna $k \geq 0$.

Je-li navíc počáteční přiblížení (x^0, y^0, s^0) voleno jako v (4.17) a (4.18), existuje kladná konstanta $\tilde{\delta}$ nezávislá na n taková, že

$$\tilde{\alpha} \geq \frac{\tilde{\delta}}{n^2}. \tag{4.25}$$

Důkaz: Podle (4.21) získáme

$$(\Delta x)^T \Delta s \; = \left(D^{-1}\Delta x\right)^T (D\Delta s\;) \leq ||D^{-1}\Delta x||.||D\Delta s\;|| \leq C_1^2 \mu. \tag{4.26}$$

Podobně

$$|\Delta x_i \Delta s_i| = |D_{ii}^{-1}\Delta x_i||D_{ii}\Delta s_i| \leq ||D^{-1}\Delta x||.||D\Delta s\ || \leq C_1^2\mu \tag{4.27}$$

(zde a dále index k vynecháváme). Na základě posledního řádku soustavy (4.6) odvodíme dvě důležité rovnosti. Sečtením všech n složek výrazu získáme

$$\begin{aligned} s^T\Delta x + x^T\Delta s\ &= e^T\left(S\ \Delta x + X\Delta s\ \right) \\ = \quad e^T\left(-XSe + \sigma\mu e\right) &= \left(\sigma - 1\right)x^T s. \end{aligned} \tag{4.28}$$

Vezmeme-li v úvahu pouze jednu složku, získáme

$$s_i\Delta x_i + x_i\Delta s_i = -x_i s_i + \sigma\mu. \tag{4.29}$$

Pro každou z nerovností (4.22)–(4.24) nalezněme podmínky na $\tilde{\alpha}$, při nichž příslušná nerovnost platí. Pro (4.22) je

$$\begin{aligned} \left(x + \alpha\Delta x\right)^T\left(s\ + \alpha\Delta s\ \right) &= x^T s\ + \alpha\left(\sigma - 1\right)x^T s\ + \alpha^2\Delta x^T\Delta s \\ &\geq \left(1 - \alpha\right)x^T s\ + \alpha\sigma x^T s\ - \alpha^2 C_1^2\mu \\ &\geq \left(1 - \alpha\right)x^T s\ + \left(\alpha\sigma_{min} - \frac{\alpha^2 C_1^2}{n}\right)x^T s. \end{aligned} \tag{4.30}$$

(4.22) tedy platí, je-li poslední výraz nezáporný, což je splněno pro

$$\alpha \leq \frac{n\sigma_{min}}{C_1^2}. \tag{4.31}$$

Poznamenejme, že (4.22) implikuje platnost podmínky (4.3), poněvadž pro

$$r_b = b - A\ x(\alpha), \qquad r_c = A^T y(\alpha) + s(\alpha) - c,$$

je

$$\frac{||\left(r_b(\alpha), r_c(\alpha)\right)||}{\mu(\alpha)} \leq \frac{\left(1 - \alpha\right)||(r_b, r_c)||}{\mu(\alpha)} \leq \frac{||(r_b, r_c)||}{\mu} \leq \beta\frac{||(r_b^0, r_c^0)||}{\mu_0}.$$

Pro důkaz (4.23) užijeme (4.27), (4.29) a fakt, že $x_i s_i \geq \gamma\mu$. Platí tedy

$$\begin{aligned} \left(x_i + \alpha\Delta x_i\right)\left(s_i + \alpha\Delta s_i\right) &\geq x_i s_i\left(1 - \alpha\right) + \alpha\sigma\mu - \alpha^2 C_1^2\mu \\ &\geq \gamma\left(1 - \alpha\right)\mu + \alpha\sigma\mu - \alpha^2 C_1^2\mu. \end{aligned} \tag{4.32}$$

Na druhé straně můžeme, jako v (4.30), ukázat, že

$$\frac{1}{n}\left(x + \alpha\Delta x\right)^T\left(s\ + \alpha\Delta s\ \right) \leq \left(1 - \alpha\right)\mu + \alpha\sigma\mu + \alpha^2 C_1^2\frac{\mu}{n}. \tag{4.33}$$

Převeďme oba výrazy z (4.23) na jednu stranu a užijme (4.32) a (4.33). Dostáváme

$$\begin{aligned}
&(x_i + \alpha\Delta x_i)(s_i + \alpha\Delta s_i) - \frac{\gamma}{n}(x + \alpha\Delta x)^T (s\ + \alpha\Delta s\ ) \\
\geq\ & \alpha\sigma(1-\gamma)\mu - \left(1 + \frac{\gamma}{n}\right)\alpha^2 C_1^2 \mu \geq \alpha\sigma_{min}(1-\gamma)\mu - 2\alpha^2 C_1^2 \mu.
\end{aligned}$$

Nerovnost (4.23) platí, je-li poslední výraz nezáporný, což je pro

$$\alpha \leq \frac{\sigma_{min}(1-\gamma)}{2C_1^2}. \tag{4.34}$$

Konečně převeďme oba výrazy z (4.24) na jednu stranu, užijme skutečnosti $\sigma \leq \sigma_{max} \leq 0.5$ a vztahu (4.33) a pišme

$$\begin{aligned}
&\frac{1}{n}(x + \alpha\Delta x)^T (s\ + \alpha\Delta s\ ) - (1 - 0.01\alpha)\mu \\
\leq\ & (1-\alpha)\mu + \alpha\sigma\mu + \alpha^2 C_1^2 \frac{\mu}{n} - (1 - 0.01\alpha)\mu \\
\leq\ & -0.99\alpha\mu + 0.5\alpha\mu + \alpha^2 C_1^2 \mu \\
=\ & -0.49\alpha\mu + \alpha^2 C_1^2 \mu.
\end{aligned}$$

Pro

$$\alpha \leq \frac{0.49}{C_1^2} \tag{4.35}$$

je poslední výraz nekladný a (4.24) tedy platí.
Na základě (4.31), (4.34) a (4.35) můžeme závěrem říct, že podmínky (4.22)–(4.24) platí, je-li $\alpha \in \langle 0, \tilde{\alpha}\rangle$, kde

$$\tilde{\alpha} = \min\left(\frac{n\sigma_{min}}{C_1^2}, \frac{\sigma_{min}(1-\gamma)}{C_1^2}, \frac{0.49}{C_1^2}, 1\right).$$

Druhou část tvrzení — omezení (4.25) — lze dokázat obdobně. □

Věta 16 *Posloupnost $\{\mu_k\}$, generovaná algoritmem IPF, konverguje Q-lineárně k nule a posloupnost norem reziduí $\{\| (r_b^k, r_c^k) \|\}$ konverguje R-lineárně k nule.*

Důkaz: Q-lineární konvergence posloupnosti $\{\mu_k\}$ plyne bezprostředně ze vztahů (4.15) a (4.25). Pro rezidua získáme vztah

$$\| (r_b^k, r_c^k) \| \leq \mu_k \beta \| (r_b^0, r_c^0) \| / \mu_0 \tag{4.36}$$

Posloupnost norem reziduí je shora omezená Q-lineárně konvergující posloupností $\{\mu_k\}$, sama tedy konverguje R-lineárně. □

4.3 Hromadné body posloupnosti iterací

V kapitole 3 jsme ukázali, že posloupnost složek (x^k, s^k) iterací, vytvořených metodami path–following, je omezená a její hromadné body tvoří složky (x^*, s^*) primárně–duálního řešení úlohy lineárního programování. Pro metodu IPF uvedeme obdobná tvrzení.

Připomeňme ještě, že rozdělení množiny indexů $\{1, \ldots, n\}$ na podmnožiny $\mathcal{B}$ a $\mathcal{N}$ je definováno jako ve (1.13), (1.14)

Věta 17 *Buď $\{(x^k, y^k, s^k)\}$ posloupnost iterací generovaných algoritmemIPF. Pak existuje konstanta C_2 taková, že pro každé k dostatečně velké platí*

$$0 < x_i^k \leq \mu_k / C_2 \quad (i \in \mathcal{N}), \qquad 0 < s_i^k \leq \mu_k / C_2 \quad (i \in B), \tag{4.37}$$

$$s_i^k \geq C_2 \gamma \quad (i \in B), \qquad x_i^k \geq C_2 \gamma \quad (i \in \mathcal{N}). \tag{4.38}$$

Tedy každý hromadný bod posloupnosti $\{(x^k, s^k)\}$ můžeme užít pro konstrukci primárně–duálního, striktně komplementárního řešení úlohy lineárního programování.

Důkaz: Pro důkaz nerovností (4.37) a (4.38) odkazuji na knihu S.Wright [2]. Zde dokážeme pouze poslední část tvrzení.
Předpokládejme, že $\mathcal{K}$ je taková posloupnost, pro niž

$$\lim_{k \in \mathcal{K}} (x^k, s^k) = (x^*, s^*).$$

Podobně jako u věty 15 získáme po přechodu k limitě vztahy

$$Ax^* = b \quad , \quad (x^*, s^*) \geq 0 \quad , \quad (x^*)^T s^* = 0.$$

Striktní komplementarita plyne okamžitě z (4.38). Dále víme

$$\lim_{k \in \mathcal{K}} r_c^k = \lim_{k \in \mathcal{K}} \left(s^k - c + A^T y^k \right) = 0$$

a tedy

$$\operatorname{dist} \left(s^* - c, \operatorname{Range}(A^T) \right) = 0.$$

Poněvadž $\operatorname{Range}(A^T)$ je uzavřená, je $s^* - c \in \operatorname{Range}(A^T)$ a tedy existuje y^* tak, že $s^* - c = A^T y^*$. (x^*, y^*, s^*) je potom striktně komplementárním řešením úlohy lineárního programování. □

K tomu, abychom dokázali omezenost posloupnosti $\{(x^k, s^k)\}$ je nutné předpokládat, že množina ostře přípustných řešení $\mathcal{F}^0$ je neprázdná. Algoritmy uvedené v kapitole 3 nebyly pro případ $\mathcal{F}^0 = \emptyset$ vůbec definovány, ovšem algoritmus IPF pro tento případ existuje a dokonce platí konvergenční věta 16.

Věta 18 *Nechť $\{(x^k, y^k, s^k)\}$ je posloupnost iterací, generovaná algoritmem IPF a nechť množina ostře přípustných řešení $\mathcal{F}^0$ je neprázdná. Pak posloupnost $\{(x^k, s^k)\}$ je omezená a má tedy alespoň jeden hromadný bod.*

K důkazu této věty je potřeba dokázat navíc některá tvrzení, která se přímo nevztahují k metodám vnitřního bodu. Nebudu jej zde proto uváďět. Důkaz věty 18 i důkazy jmenovaných tvrzení lze opět nalézt v knize S.Wright [2].

Dodatek A

A.1 Primární metody

Uvažujme problém (P)

$$\{\min c^T x\,;\; Ax = b\; x \geq 0\}$$

a převeďme ho na problém

$$\min \left(c^T x - \tau \sum_{i=1}^{n} \log x_i \right)$$
$$\text{na množině} \quad \{x \in R^n \,:\, Ax = b\} \tag{A.1}$$

Lagrangeova funkce pro (A.1) má tvar

$$L(x, y, \tau) := c^T x - \tau \sum \log x_i - y^T (Ax - b) \tag{A.2}$$

a podmínky prvního řádu pro (A.2) jsou

$$\nabla_x L = c - \tau X^{-1} e - A^T y = 0, \tag{A.3}$$
$$\nabla_y L = -Ax + b = 0, \tag{A.4}$$

kde X je obvyklé značení pro diag(x).

Předpokládáme-li že existuje ostře přípustné řešení úlohy (P) x^k (tj. $x^k > 0,\ Ax^k = b$) a aplikujeme-li Newtonovu metodu na soustavu (A.3), (A.4) získáme směr Δx^k tvaru

$$\Delta x^k = -\frac{1}{\tau^k} X^k P X^k c + X^k P e, \tag{A.5}$$

kde

$$P = I - X^k A^T (A(X^k)^2 A^T)^{-1} A X^k.$$

Nové přiblížení x^{k+1} je potom

$$x^{k+1} := x^k + \alpha^k \Delta x^k \tag{A.6}$$

pro vhodnou délku kroku α^k. Hodnotu barierového parametru pro následující iteraci τ^{k+1} definujeme jako $\vartheta \tau^k$, kde $0 < \vartheta < 1$.

Jiný způsob je namísto barierové metody užít primární afinní metodu. Směr Δx^k má potom tvar

$$\Delta x^k = -X^k P X^k c. \tag{A.7}$$

Primární metody zachovávají v každém kroku hodnotu x^k kladnou, na hodnoty duálních proměnných žádná omezení nekladou.

Více o primárních metodách lze nalézt například v pracech P.E.Gill & W.Murray & D.B.Ponceleon & M.A.Saunders [15], E.R.Barnes [16], R.J.Vanderbei & M.S.Meketon & B.A.Freedman [17].

A.2 Duální metody

Podobně jako jsme u primárních metod uvažovali pouze primární úlohu, uvažujme nyní pouze problém (D)

$$\{\max b^T y\,;\; A^T y + s \;= c,\; s \;\geq 0\},$$

který je ekvivalentní problému

$$\{\min -b^T y\,;\; A^T y + s \;= c,\; s \;\geq 0\}$$

a převeďme ho, podobně jako v předchozím odstavci, na problém

$$\min(-b^T y - \tau \sum_{i=1}^{n} \log s_i)$$

$$\text{na množině} \quad \{(y,s) \in R^m \times R^n \;:\; A^T y + s \;= c\} \tag{A.8}$$

Nyní upravme (A.8) do tvaru

$$\max \left(b^T y - \tau \sum_{i=1}^{n} \log(c_i - a_i^T y) \right) \tag{A.9}$$

kde a_j je j-tý sloupek matice A. Podmínky prvního řádu pro (A.9) jsou

$$b - \tau A \; S^{-1} e = 0, \tag{A.10}$$

kde S je diagonální matice typu $n \times n$, jejíž prvky jsou $s_i = c_i - a_i^T y$. Jeden krok Newtonovy metody dává

$$\Delta y^k = \frac{1}{\tau^k}(A(S^k)^{-2}A^T)^{-1}b - (A(S^k)^{-2}A^T)^{-1}AS^{-1}e, \tag{A.11}$$

kde první výraz v (A.11) způsobí přiblížení k optimu a druhý výraz zlepší centralitu iterace v duálním prostoru.

Místo barierové metody můžeme opět užít duální afinní metodu. Směr Δy^k má potom tvar

$$\Delta y^k = (A(S^k)^{-2}A^T)^{-1}b. \tag{A.12}$$

Více lze nalézt například v P.E.Gill & W.Murray & D.B.Ponceleon & M.A.Saunders [15], I.Adler et al. [18].

Dodatek B

B.1 Důkaz věty o dualitě

Než větu a dualitě dokážeme, uveďme několik potřebných tvrzení. Nejprve definujeme úlohu speciálního tvaru

$$\min\{a^T x; Cx \geq -a,\ x \geq 0\}, \tag{SP}$$

kde C je matice typu n x n, $x, a \in R^n$, $a \geq 0$. O matici C navíc předpokládejme, že má vlastnost

$$C^T = -C. \tag{B.1}$$

Lze snadno nahlédnout, že pro každé $x \in R^n$ je

$$x^T Cx = 0 \tag{B.2}$$

Odpovídající duální úloha má tvar

$$\max\{-a^T y; C^T y \leq a,\ y \geq 0\}, \tag{SD}$$

kde $y \in R^n$.

Úloha (SP) je tzv. samoduální úloha, t.j. taková, pro niž má duální úloha stejný tvar jako úloha primární. Protože má matice C vlastnost (B.1), je i množina přípustných řešení primární úlohy (SP) stejná jako množina přípustných řešení duální úlohy (SD).

Lemma 12 *(SP) i (SD) jsou přípustné a pro obě z nich je optimálním řešením nulový vektor.*

Důkaz: Protože $a \geq 0$, je nulový vektor přípustný jak pro primární, tak pro duální úlohu. Pro každé přípustné řešení primární úlohy navíc platí $0 = x^T Cx \geq -a^T x$ a tedy $a^T x \geq 0$; analogicky $a^T y \geq 0$ pro každé přípustné řešení duální úlohy. Nulový vektor je tedy optimálním řešením jak primární, tak duální úlohy. □

Důsledek 3 *Buď x přípustné pro (SP) a definujme $s := Cx + a$. Pak x je optimální tehdy a jen tehdy, je-li $x^T s = 0$.*

Důkaz:

$$a^T x = s^T x - x^T C^T x = s^T x. \square$$

Abychom mohli dokázat větu o dualitě pro obecné úlohy, dokážeme ji nejprve pro úlohy (SP),(SD). Poněvadž obě úlohy (SP) i (SD) mají stejný tvar i stejnou množinu přípustných řešení, budeme v dalším používat pouze značení (SP).

Definujme množinu přípustných řešení pro úlohu (SP)

$$\mathcal{SP} := \{(x,s) : Cx - s \ = -a,\ x \geq 0,\ s \ \geq 0\},$$

množinu ostře přípustných řešení pro úlohu (SP)

$$\mathcal{SP}^0 := \{(x,s) : Cx - s \ = -a,\ x > 0,\ s \ > 0\},$$

a množinu optimálních řešení pro úlohu (SP)

$$\Omega_{SP} := \{(x,s) : Cx - s \ = -a,\ x^T s = 0,\ x \geq 0,\ s \ \geq 0\}.$$

Lemma 13 *Buď $D \subseteq R^n$ otevřená konvexní množina a buď $f : D \to R$ konvexní diferencovatelná funkce. Pak funkce f nabývá (na D) svého minima v bodě $x \in D$ tehdy a jen tehdy, je-li $\nabla f(x) = 0$.*

Lemma 14 *Buďte dány $\tau \in R, \tau > 0$, a $p \in R^n, p > 0$. Funkce $h(x) := p^T x - \tau \sum_{i=1}^{n} \log x_i$ má jednoznačně určené minimum.*

Důkaz lemmatu 14 lze nalézt např. v B.Jansen [19].

Pro $\tau > 0$ definujme bariérovou funkci $f_\tau : R^n_+ \to R$ předpisem

$$f_\tau(x) := a^T x - \tau \left(\sum_{i=1}^{n} \log x_i + \sum_{i=1}^{n} \log(c_i.x + a_i) \right),$$

kde c_i znamená i-tý řádek matice C.

Lemma 15 *Buď $\tau > 0$. Následující tvrzení jsou ekvivalentní*

(i) Funkce f_τ má (jednoznačné) minimum.

(ii) Existují vektory $x, s \in R^n$, pro něž

$$\begin{aligned} Cx - s \ = -a, \quad x \geq 0, \quad s \ \geq 0 \\ Xs = \tau e \end{aligned} \tag{B.3}$$

Platí-li jedno z uvedených tvrzení, pak f_τ nabývá svého minima v bodě x právě když x a s splňují podmínky (B.3).

Důkaz: Poznamenejme nejprve, že když (x,s) řeší soustavu (B.3), jsou složky x_i, s_i kladné.(Vyplývá z druhé rovnice.) Je lehce ověřitelné, že $f_\tau(x)$ je ostře konvexní a nabývá tedy svého minima v nejvýše jednom bodě. Protože definiční obor funkce f_τ je otevřená množina, vyplývá z lemmatu 13, že f_τ má v bodě x minimum právě tehdy, když $\nabla f_\tau = 0$, tj.

$$a - \tau X^{-1} e - \tau C^T S^{-1} e = 0, \tag{B.4}$$

kde $X = diag(x)$, $S = diag(s)$, $e = (1, \ldots, 1)^T$. Užijeme-li vztahů $s = Cx + a$ a $C^T = -C$, můžeme (B.4) přepsat jako

$$\tau X^{-1}e - s = C(\tau S^{-1}e - x),$$

čili

$$0 = (C - X^{-1}S)S^{-1}(\tau e - Xs).$$

Poněvadž C má vlastnost (B.1) a matice $X^{-1}S$ a S^{-1} jsou diagonální a pozitivně definitní, platí poslední vztah právě tehdy, když $Xs = \tau e$.□

Předpokládejme, že množina $\mathcal{SP}^0$ je neprázdná a zvolme $(x^{(0)}, s^{(0)}) \in \mathcal{SP}^0$ Podle (B.2) je pro každé $(x, s) \in \mathcal{SP}$

$$(x - x^{(0)})^T(s - s^{(0)}) = (x - x^{(0)})^T C(x - x^{(0)}) = 0. \tag{B.5}$$

Ekvivalentně

$$(x^{(0)})^T s + (s^{(0)})^T x = x^T s + (x^{(0)})^T (s^{(0)}) = a^T x + a^T x^{(0)},$$

neboli

$$a^T x = (x^{(0)})^T s + (s^{(0)})^T x - a^T x^{(0)}. \tag{B.6}$$

Definujeme-li nyní funkci $g_\tau : R^n_+ \times R^m_+ \to R$ předpisem

$$g_\tau(x, s) := (x^{(0)})^T s + (s^{(0)})^T x - \tau \left(\sum_{i=1}^{n} \log x_i + \sum_{i=1}^{n} \log s_i \right),$$

je pro každé $(x, s) \in \mathcal{SP}^0$

$$g_\tau(x, s) = f_\tau(x) + a^T(x^{(0)}).$$

Funkce $g_\tau(x, s)$ a $f_\tau(x)$ se na množině $\mathcal{SP}^0$ liší pouze o konstantu.

Věta 19 *Buď $\tau > 0$. Následující tvrzení jsou ekvivalentní*

(i) Množina $\mathcal{SP}^0$ je neprázdná.

(ii) Funkce f_τ má (jednoznačné) minimum.

(iii) Systém (B.3) má (jednoznačné) řešení.

Důkaz: Ekvivalence *(ii)* <=> *(iii)* je obsahem lemmatu 15 *(iii)* => *(i)* je zřejmé. Zbývá tedy dokázat, že (i) implikuje (ii). Předpokládejme, že platí (i) a buď $(x^{(0)}, s^{(0)}) \in \mathcal{SP}^0$. Základní myšlenka důkazu je tato: protože platí (B.6), je minimalizace $f_\tau(x)$ na množině R^n_+ ekvivalentní minimalizaci $g_\tau(x, s)$ ma množině $\mathcal{SP}^0$. Stačí tedy dokázat, že funkce g_τ nabývá svého minima na množině $\mathcal{SP}^0$. K tomu stačí užít vlastností definičního oboru funkce g_τ a omezenosti množin

$$\mathcal{L}_K := \{(x,s) : g_\tau(x,s) \leq K\ \}.\Box$$

Podrobněji je důkaz rozebrán v B.Jansen [19].

Pro každé kladné τ označme $x(\tau) := \arg\min f_\tau(x)$ a definujme $s(\tau) := Cx(\tau) + a$.

Lemma 16 *Buď $\bar{\tau} > 0$. Pak je množina $\{(x(\tau), s(\tau)) : 0 < \tau \leq \bar{\tau}\}$ omezená.*

Důkaz: Buďte $(x^{(0)}, s^{(0)}) \in \mathcal{SP}^0$. Užijeme-li vlastnosti (B.5) a faktu, že pro $x(\tau)$ platí (B.3), získáme pro každé i, $1 \leq i \leq n$, vztah

$$\begin{aligned} s_i^{(0)} x_i(\tau) &\leq (x^{(0)})^T s(\tau) + (s^{(0)})^T x(\tau) \\ &= x(\tau)^T s(\tau) + (x^{(0)})^T s^{(0)} \\ &= n\tau + (x^{(0)})^T s^{(0)} \\ &\leq n\bar{\tau} + (x^{(0)})^T s^{(0)}. \end{aligned} \tag{B.7}$$

A tedy můžeme říct, že $x_i(\tau) \leq (n\bar{\tau} + (x^{(0)})^T s^{(0)})/s_i^{(0)}$. Množina $\{x(\tau) : 0 < \tau \leq \bar{\tau}\}$ je omezená. Podobně pro $\{s(\tau) : 0 < \tau \leq \bar{\tau}\}$.$\Box$

Věta 20 *Je-li množina $\mathcal{SP}^0$ neprázdná, existuje $(x^*, s^*) \in \Omega_{SP}$ takové, že $x^* + s^* > 0$.*

Důkaz: Buď $\{\tau^k\}$ posloupnost kladných čísel taková, že $\lim_{k\to\infty} \tau^k = 0$. Podle lemmatu 16 je množina $\{(x(\tau^k), s(\tau^k))\}$ omezená, tedy obsahuje alespoň jednu konvergentní podposloupnost. Její limitu označme (x^*, s^*). Poněvadž $(x^*, s^*) \in \mathcal{SP}$ a $x(\tau^k)^T s(\tau^k) = n\tau^k \to 0$ je $(x^*)^T s^* = 0$ a tedy $(x^*, s^*) \in \Omega_{SP}$. Ukažme, že (x^*, s^*) je striktně komplementární. Podle (B.5) je

$$(x(\tau^k) - x^*)^T (s(\tau^k) - s^*) = 0.$$

Užijeme-li $x(\tau^k)^T s(\tau^k) = n\tau^k$ a $(x^*)^T s^* = 0$, lze tento vztah přepsat jako

$$\sum_{i\in\mathcal{B}} x_i^* s_i(\tau^k) + \sum_{i\in\mathcal{N}} x_i(\tau^k) s_i^* = n\tau^k$$

Vydělme obě strany τ^k a užijme skutečnosti $x_i(\tau^k)s_i(\tau^k) = \tau^k$. Získáme

$$\sum_{i\in\mathcal{B}} \frac{x_i^*}{x_i(\tau^k)} + \sum_{i\in\mathcal{N}} \frac{s_i^*}{s_i(\tau^k)} = n$$

Přechodem $k \to \infty$ zjistíme, že hodnota první sumy je rovna počtu nenulových složek vektoru x^*, podobně hodnota druhé sumy je rovna počtu nenulových složek vektoru s^*. Z toho plyne, že (x^*, s^*) je striktně komplementární. $\Box$

Důsledek 4 *Z důkazu věty vyplývá, že z bodů na centrální cestě $\{(x(\tau), s(\tau)) : \tau > 0\}$ lze vybrat podposloupnost, konvergující k striktně komplementárnímu optimálnímu řešení.*

Výsledky předchozích tvrzení nyní použijeme k důkazu věty o dualitě. Větu dokážeme pro dvojici úloh

$$\min\{c^T x;\ Ax = b,\ x \geq 0\} \tag{$\hat{\mathrm{P}}$}$$

$$\max\{b^T y;\ A^T y \leq c\}. \tag{$\hat{\mathrm{D}}$}$$

Nejprve poznamenejme, že pro primární a duální úlohu formulovanou pomocí nerovností znamená striktní komplementarita toto:

Definice: Dvojice (x^*, y^*) je striktně komplementární, je-li

- x^* přípustné pro úlohu $(\hat{P})$
- y^* přípustné pro úlohu $(\hat{D})$

a navíc

$$\begin{aligned} (Ax^* - b)^T y^* = (c - A^T y^*)^T x^* &= 0, \\ y^* + (Ax^* - b) &> 0, \\ x^* + (c - A^T y^*) &> 0. \end{aligned}$$

Vytvoříme samoduální problém tvaru (SP) s maticí, která má vlastnost (B.1).

Zvolme nejprve libovolné vektory $x^{(0)}, r^{(0)} \in R^n_+,\ y^{(0)}, u^{(0)} \in R^m_+$ a libovolná kladná čísla ϑ_0, τ^0, λ^0, ν^0. Dále definujme $\bar{c}$, $\bar{b}$, α, β následujícím způsobem:

$$\begin{aligned} \bar{b} &:= (\lambda^0 b - A\ x^{(0)} + r^{(0)})/\vartheta^0, \\ \bar{c} &:= (\lambda^0 c - A^T y^{(0)} - u^{(0)})/\vartheta^0, \\ \alpha &:= (c^T x^{(0)} - b^T y^{(0)} + \tau^0)/\vartheta^0, \end{aligned}$$

$$\begin{aligned} \beta &:= \alpha\lambda^0 + \bar{b}^T y^{(0)} - \bar{c}^T x^{(0)} + \nu^0 \\ &= ((y^{(0)})^T r^{(0)} + (x^{(0)})^T u^{(0)} + \tau^0\lambda^0)/\vartheta^0 + \nu^0. \end{aligned}$$

Je-li $x^{(0)}$ ostře přípustné pro úlohu $(\hat{P})$ a je-li $r^{(0)} := Ax^{(0)} - b$, pak pro $\lambda^0 = \vartheta^0 = 1$ je $\bar{b} = 0$. Je-li $y^{(0)}$ ostře přípustné pro úlohu $(\hat{D})$ a je-li $u^{(0)} := c - A^T y^{(0)}$, pak pro $\lambda^0 = \vartheta^0 = 1$ je $\bar{c} = 0$. Tedy $\bar{b}$ a $\bar{c}$ udávají "míru nepřípustnosti" zvolených vektorů $x^{(0)}$, $r^{(0)}$, $y^{(0)}$, $u^{(0)}$.

Definujme nyní problém

$$min\beta\vartheta \tag{$\hat{SP}$}$$

na množině takových $(x, y, \vartheta, \lambda)$, které vyhovují soustavě

$$\begin{bmatrix} 0 & A & \bar{b} & -b \\ -A^T & 0 & -\bar{c} & c \\ -\bar{b} & \bar{c}^T & 0 & -\alpha \\ b^T & -c^T & \alpha & 0 \end{bmatrix} \begin{bmatrix} y \\ x \\ \vartheta \\ \lambda \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \\ -\beta \\ 0 \end{bmatrix} \tag{B.8}$$

$$(y, x, \vartheta, \lambda) \geq 0,$$

matice soustavy je typu $(m+n+1+1) \times (m+n+1+1)$.

Lze ověřit, že vektor $x = x^{(0)}$, $y = y^{(0)}$, $\vartheta = \vartheta^0$, $\lambda = \lambda^0$ je přípustným řešením $(\hat{SP})$ a tedy $\mathcal{SP}^0 \neq \emptyset$. Koeficienty účelové funkce jsou nezáporné. Na úlohu $(\hat{SP})$ lze tudíž aplikovat výsledky předchozích vět a lemmat. Uveďme znovu větu o dualitě.

Věta 21 *(o dualitě): Pro úlohy $(\hat{P})$ a $(\hat{D})$ platí jedna z následujících alternativ*

(i) $(\hat{P})$ i $(\hat{D})$ jsou přípustné a obě mají optimální řešení $x^* \in \Omega_P$, $(y^*, s^*) \in \Omega_D$.

(ii) $(\hat{P})$ je nepřípustná a účelová funkce $(\hat{D})$ je (shora) neomezená.

(iii) $(\hat{D})$ je nepřípustná a účelová funkce $(\hat{P})$ je (zdola) neomezená.

(iv) Obě úlohy $(\hat{P})$ i $(\hat{D})$ jsou nepřípustné.

Důkaz: Problém $(\hat{SP})$ je samoduální problém s maticí, která má vlastnost (B.1), koeficienty účelové funkce jsou nezáporné a množina $\mathcal{SP}^0$ je neprázdná. Podle věty 20 existuje striktně komplementární řešení $(x^*, y^*, \vartheta^*, \lambda^*)$. Z lemmatu 12 zároveň víme, že $\vartheta^* = 0$, protože $\beta \geq \nu^0 > 0$. Nyní můžou nastat dvě možnosti:
je-li $\lambda^* > 0$, pak $\bar{x}^* := x^*/\lambda^*$ resp. $\bar{y}^* := y^*/\lambda^*$ jsou přípustná řešení pro úlohu $(\hat{P})$ resp. $(\hat{D})$ a tvoří jejich ostře komplementární řešení. Tedy platí případ (i).

Je-li, na druhé straně,$\lambda^* = 0$, je potom $Ax^* \geq 0, x^* \geq 0, A^T y^* \leq 0, y^* \geq 0$ a $b^T y^* - c^T x^* > 0$.Je-li $b^T y^* > 0$, je $(\hat{P})$ nepřípustná. (Kdyby totiž existovalo přípustné řešení primární úlohy $\bar{x}$, bylo by $0 \geq \bar{x}^T A^T y^* \geq b^T y^*$, což je spor.) Okamžitě také vyplývá, že je-li v pro tento případ $(\hat{D})$ přípustná, je její účelová funkce neomezená. Je-li $c^T x^* < 0$, je $(\hat{D})$ nepřípustná, neboť pro $\bar{y}$ přípustné pro dualní úlohu je $0 \leq \bar{y}^T A\, x^* \leq c^T x^*$,což je spor. Je-li v tomto případě $(\hat{P})$ přípustná, je její účelová funkce neomezená. Obdobně lze ukázat, že pro $b^T y^* > 0$ a $c^T x^* < 0$ jsou obě úlohy $(\hat{P})$ i $(\hat{D})$ nepřípustné. □

Obdobnou cestou můžeme dokázat i Goldmanovu–Tuckerovu větu (věta 8) pro obecnou úlohu.

B.2 Lineární algebra

Velkou část numerických výpočtů u primárně–duálních metod tvoří vyřešení soustavy (2.10) resp.(4.6). Matice těchto soustav je obvykle hodně velká a řídká a to už z toho důvodu, že sama matice A bývá pro většinu úloh lineárního programování hodně velká a řídká. Její speciální struktura nám však dovoluje přepsat ji do symetrického tvaru a výsledná soustava pak bude řešitelná snadněji a rychleji.

Uvažujme např. soustavu (4.6). Vyjádřením Δs z třetího výrazu a dosazením tohoto vyjádření do druhého výrazu získáme soustavu

$$\begin{bmatrix} 0 & A \\ A^T & -D^{-2} \end{bmatrix} \begin{bmatrix} \Delta y \\ \Delta x \end{bmatrix} = \begin{bmatrix} -r_b \\ -r_c + s - \sigma\mu X^{-1} e \end{bmatrix}, \tag{B.9}$$

$$\Delta s = -s + \sigma\mu X^{-1} e - X^{-1} S\, \Delta x. \tag{B.10}$$

Matice D je definovaná předpisem (4.20). Tento tvar soustavy pro výběr směru je znám pod názvem augmented system. Poněvadž $x > 0$ i $s > 0$, je matice D^{-2} diagonální a regulární.

Dalším krokem může být vyeliminování výrazu Δx z druhé rovnice (B.9) a jeho dosazení do první rovnice a do (B.10). Tím získáme

$$\begin{aligned} A\ D^2 A^T \Delta y &= -r_b + A\ \left(S^{-1} X r_c + x - \sigma\mu S^{-1} e\right), && \text{(B.11)} \\ \Delta s &= -r_c - A^T \Delta y, && \text{(B.12)} \\ \Delta x &= -x + \sigma\mu S^{-1} e - S^{-1} X \Delta s. && \text{(B.13)} \end{aligned}$$

Tento tvar se často nazývá normal equations.

B.3 Jednoznačná řešitelnost Newtonových soustav

V tomto odstavci dokážeme, že, je-li pro přiblížení (x, y, s) vektor $(x, s) > 0$, jsou soustavy (2.10) a (4.6) jednoznačně řešitelné ve složkách $(\Delta x, \Delta s)$. Abychom ověřili tuto skutečnost, dokažme, že pro řešení $(\Delta x, \Delta y, \Delta s)$ homogenní soustavy

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \qquad \text{(B.14)}$$

musí platit $\Delta x = 0, \Delta s\ = 0$. Z prvních dvou (blokových) řádků soustavy vyplývá

$$\Delta x^T \Delta s\ = -\Delta x^T A^T \Delta y = -(A\Delta x)^T \Delta y = 0$$

Buď D diagonální matice definovaná v (4.20), jejíž diagonální prvky jsou kladné. Vynásobíme-li poslední řádek soustavy (B.14), výrazem $(XS)^{-1/2}$ získáme

$$D^{-1}\Delta x + D\Delta s\ = 0.$$

Nyní užijme vztahu $(\Delta x)^T \Delta s\ = 0$. Platí

$$\begin{aligned} 0 &= \|D^{-1}\Delta x + D\Delta s\ \|_2^2 \\ &= \|D^{-1}\Delta x\|_2^2 + 2(\Delta x)^T \Delta s\ + \|D\Delta s\ \|_2^2 \\ &= \|D^{-1}\Delta x\|_2^2 + \|D\Delta s\ \|_2^2. \end{aligned}$$

Tedy $D^{-1}\Delta x = 0$ a $D\Delta s\ = 0$, z čehož plyne $\Delta x = 0$ a $\Delta s\ = 0$, což jsme měli dokázat.

Má-li navíc matice A plnou hodnost ($rank A\ = m$), je řešení $(\Delta x, \Delta y, \Delta s)$ jednoznačné i ve složce Δy. Jestliže totiž do prvního řádku dosadíme $\Delta s\ = 0$, získáme $A^T \Delta y = 0$ a tudíž $\Delta y = 0$.□

B.4 Typy konvergence uvažované v textu

Definice Nechť posloupnost aproximací $\{x_i\}$ konverguje k bodu x^*. Konvergence je řádu m, jestliže existuje index $k \in N$ a konstanta A tak, že

$$\|x_{i+1} - x^*\| \leq A \, \|x_i - x^*\|^m \qquad \forall i \geq k$$

Je-li $m = 1$ řekneme, že posloupnost $\{x_i\}$ konverguje k bodu x^* lineárně, je-li $m = 2$, řekneme, že konverguje kvadraticky .

Definice Nechť posloupnost aproximací $\{x_i\}$ konverguje k bodu x^*. Jestliže existuje index $k \in N$ a hodnoty $0 < M_k < \infty$ a $0 < q < 1$ tak, že

$$\|x_i - x^*\| \leq M_k q^{i-1} \|x_k - x^*\| \qquad \forall i \geq k$$

tj.

$$\frac{\|x_i - x^*\|}{\|x_k - x^*\|} \leq M_k q^{i-1} \qquad \forall i \geq k$$

řekneme, že posloupnost $\{x_i\}$ konverguje k bodu x^* R-lineárně.

Definice Nechť posloupnost aproximací $\{x_i\}$ konverguje k bodu x^*. Jestliže existuje index $k \in N$ a konstanta $0 < q < 1$ tak, že

$$\frac{\|x_{i+1} - x^*\|}{\|x_i - x^*\|} \leq q \qquad \forall i \geq k$$

řekneme, že posloupnost $\{x_i\}$ konverguje k bodu x^* Q-lineárně.

Definice Řekneme, že posloupnost aproximací $\{x_i\}$ konverguje k bodu x^* Q-superlineárně, jestliže

$$\lim_{i \to \infty} \frac{\|x_{i+1} - x^*\|}{\|x_i - x^*\|} = 0$$

.

Definice Nechť posloupnost aproximací $\{x_i\}$ konverguje k bodu x^*. Jestliže existuje index $k \in N$ a konstanta $M_k \in (0, \infty)$ tak, že

$$\frac{\|x_{i+1} - x^*\|}{\|x_i - x^*\|^2} \leq M_k \qquad \forall i \geq k$$

řekneme, že posloupnost $\{x_i\}$ konverguje k bodu x^* Q-kvadraticky.

Věta 22 *Nechť posloupnost aproximací $\{x_i\}$ konverguje k x^* Q-lineárně (Q-superlineárně), pak konverguje k x^* také R-lineárně (R-superlineárně).*

Dodatek C

V této části bych ráda uvedla tři konkrétní, v praxi realizované algoritmy. Ve všech třech případech se jedná o nepřípustné, primárně–duální algoritmy vnitřního bodu typu prediktor–korektor. První z nich je popsán v odstavci C.1 a další dva v odstavci C.2. U všech tří uvádím stručnou analýzu, především pro volbu délky kroku, a bez důkazů věty o složitosti algoritmů a konvergenční věty. V odstavci C.3 jsou podrobněji rozebrány naprogramované metody včetně volby vstupních parametrů a délky kroku pro jednotlivé algoritmy. V odstavci C.4 jsou potom výpisy programů.

Nejprve opět zaveďme některá značení. Podobně jako v kapitole 2 první části označme

$$F(x,y,s) := \begin{pmatrix} Ax-b \\ A^Ty+s\ -c \\ XSe \end{pmatrix}$$

$$\mathcal{F} := \{(x,y,s);\ Ax=b,\ A^Ty+s\ =c,\ (x,s)\geq 0\}$$

$$\mathcal{F}^0 := \{(x,y,s);\ Ax=b,\ A^Ty+s\ =c,\ (x,s)> 0\},$$

$$\Omega := \{(x,y,s);\ F(x,y,s)=0\ (x,s)\geq 0\},$$

Definujme navíc množinu dostatečně přesných aproximací optimálního řešení předpisem

$$\Omega_\epsilon := \{(x,y,s);\ x^Ts\leq\epsilon,\ \parallel Ax-b\parallel\leq\epsilon,\ \parallel A^Ty+s\ -c\parallel\leq\epsilon,\ (x,s)\geq 0\}$$

C.1 Metoda podle S.Mizuna

První z metod, které chci popsat v této části, jsem převzala z článku S.Mizuno [20]. Metoda je kombinací Kojimova - Megiddova - Mizunova nepřípustného algoritmu prediktor–korektor a Mizunova - Toddova - Yeova (přípustného) algoritmu prediktor–korektor, uvedeného v kapitole 3 první části. Algoritmus má opět polynomiální složitost (pro řešení úloh (P) a (D) potřebuje polynomiální čas). Tuto skutečnost zaručíme speciální volbou počátečního přiblížení, délky kroku a kritérií pro ukončení algoritmu. Nemusíme

tedy à priori požadovat přípustnost úlohy nebo existenci optimálního řešení. Fakt, že předem většinou nevíme, je-li úloha přípustná či nepřípustná, omezená či neomezená totiž způsobuje většinu obtíží v analýze nepřípustných metod vnitřního bodu. "Řešit" úlohu lineárního programování potom znamená buď

(a) nalézt aproximaci (x^k, y^k, s^k) z množiny $\mathcal{F}_\epsilon$

nebo

(b) zjistit, že úloha nemá přípustné resp. optimální řešení.

Nechť, jako v předchozím, je A matice typu $m \times n, b \in R^m, c \in R^n$.
Uvažujme úlohu LP ve standardním tvaru

$$\begin{aligned} &\min c^T x \\ &\text{na množině} \quad \{x \in R^n : Ax = b,\ x \geq 0\} \end{aligned} \tag{P}$$

a k ní duální úlohu

$$\begin{aligned} &\max b^T y \\ &\text{na množině} \quad \{(y, s) \in R^m \times R^n : A^T y + s = c,\ s\ \geq 0\}. \end{aligned} \tag{D}$$

Předpokládejme, že matice A má plnou hodnost (tj. $rankA\ = m$). Jako v předchozím řekneme, že bod (x, y, s) je (nepřípustný) vnitřní bod, je-li $x > 0$ a $s > 0$; bod (x, y, s) je přípustný vnitřní bod, je-li navíc $Ax = b$ a $A^T y + s\ = c$. Abychom algoritmus uvedený v této kapitole odlišili od algoritmů z následující kapitioly, označme ho Algoritmus A.

Kojimovo-Megiddovo-Mizunovo schéma

Nejprve pouze v krátkosti popišme Kojimovo-Megiddovo-Mizunovo schéma, jehož modifikací Algoritmus A vznikne. Definujme konstanty $0 < \gamma < 1, \gamma_P > 0, \gamma_D > 0, \epsilon > 0, \epsilon_P > 0, \epsilon_D > 0, \omega^* > 0$ a množinu

$$\begin{aligned} \mathcal{N} := \{(x, y, s) :\quad & x > 0, s > 0, && \text{(C.1)} \\ & x_i s_i \geq \gamma x^T s/n (i = 1, 2, \ldots, n), && \text{(C.2)} \\ & x^T s \geq \gamma_P \|Ax - b\| \text{ nebo } \|Ax - b\| \leq \epsilon_P, && \text{(C.3)} \\ & x^T s \geq \gamma_D \|A^T y + s\ - c\| \\ & \qquad\qquad \text{nebo } \|A^T y + s\ - c\| \leq \epsilon_D\} && \text{(C.4)} \end{aligned}$$

která tvoří okolí centrální cesty $\{(x_\tau, y_\tau, s_\tau);\ \tau > 0\}$. Buďte dále $0 < \beta_1 < \beta_2 < \beta_3 < 1$. V každé iteraci pak definujeme $\tau := \beta_1 \frac{x^T s}{n}$ a Newtonův směr $(\Delta x, \Delta y, \Delta s\ )$ počítáme ze soustavy

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} \Delta x^p \\ \Delta y^p \\ \Delta s^p \end{bmatrix} = - \begin{bmatrix} Ax^k - b \\ A^T y^k + s^k - c \\ X^k S^k e - \tau e \end{bmatrix}, \tag{C.5}$$

parametry β_2 a β_3 kontrolují délku primárního a duálního kroku. Za počáteční přiblížení zvolme libovolný bod $(x^0, y^0, s^0) \in \mathcal{N}$. Nyní můžeme popsat Kojimovo-Megiddovo-Mizunovo schéma.

Pro $k = 0, 1, 2 \ldots$ proveďme

Krok 1: Je-li

$$x^T s \leq \epsilon \quad \& \quad \|Ax^k - b\| \leq \epsilon_P \quad \& \quad \|A^T y + s^k - c\| \leq \epsilon_D$$

nebo

$$\|(x^k, s^k)\|_1 > \omega^*$$

pak **STOP**, jinak

Krok 2: Položme $\tau := \beta_1 \frac{x^T s}{n}$ a ze soustavy (C.5) spočtěme směr $(\Delta x, \Delta y, \Delta s\ )$.

Krok 3: Buď $\bar{\alpha}^k$ největší z $\tilde{\alpha} \leq 1$ takových , že vztahy

$$(x^k, y^k, s^k) + \alpha(\Delta x, \Delta y, \Delta s\ ) \in \mathcal{N}, \tag{C.6}$$

$$(x^k + \alpha \Delta x)^T (s^k + \alpha \Delta s) \leq (1 - \alpha(1 - \beta_2))(x^k)^T s^k \tag{C.7}$$

platí pro všechna $\alpha \in \langle 0, \tilde{\alpha} \rangle$.

Krok 4: Zvolme primární délku kroku α_p^k a duální délku kroku α_d^k tak, aby nová iterace $(x^{k+1}, y^{k+1}, s^{k+1})$ splňovala vztahy

$$(x^{k+1}, y^{k+1}, s^{k+1}) := (x^k + \alpha^k{}_p \Delta x, y^k + \alpha_d^k \Delta y, s^k + \alpha_d^k \Delta s) \in \mathcal{N}, \tag{C.8}$$

$$(x^{k+1})^T s^{k+1} \leq (1 - \bar{\alpha}^k (1 - \ \beta_3))(x^k)^T s^k \tag{C.9}$$

Kojima, Megiddo a Mizuno [24] ukázali, že algoritmus skončí v konečném počtu kroků, a zavedli hodnotu ω^* takovou, aby platilo: skončí-li algoritmus splněním podmínky $\|(x^k, s^k)\|_1 > \omega^*$, nemají úlohy (P) a (D) v jisté, předem definované, oblasti žádný přípustný bod. Hodnotu ω^* lze pro jednoduchost volit $\omega^* := \infty$; v Algoritmu A proto podmínku $\|(x^k, s^k)\|_1 > \omega^*$ vynecháme.

Zaveďme na tomto místě ještě jednu podmínku. Buď

$$\rho_0 \geq \min\{\|(u, w)\|_\infty : Au = b,\ A^T v + w = c \text{ pro nějaké } v\ \} \tag{C.10}$$

a buď $\rho \geq \rho_0$ námi zvolená konstanta. Pro ni se pak snažíme nalézt optimální řešení x^* úlohy (P) a optimální řešení (y^*, s^*) úlohy (D) tak, aby platilo

$$\|(x^*, s^*)\|_\infty \leq \rho \tag{C.11}$$

Algoritmus A

V našem případě definujme pro $\gamma \in (0, 1)$ okolí centrální cesty následovně

$$\mathcal{N}_2(\gamma) := \{(x, y, s) : x > 0,\ s > 0,\ \|XSe - \mu e\| \leq \gamma \mu\} \tag{C.12}$$

kde μ je, jako v předchozím, průměrná hodnota součinů $x_i s_i$ definovaná vztahem

$$\mu := x^T s / n.$$

Algoritmus A generuje posloupnost $\{(x^k, y^k, s^k)\}_k \in \mathcal{N}_2(\frac{1}{4})$. Protože se však jedná o metodu prediktor–korektor, založenou na Algoritmu PC z kapitoly 3 první části, leží iterace po korektorovém kroku v okolí $\mathcal{N}_2(\frac{1}{2})$.

Mějme dáno $0 < \beta_1 < \beta_2 < 1,\ 0 < \gamma_0 \leq 1,\ \gamma_1 := \frac{1}{4},\ , \rho > 0,\ r^1 := 1$, a položme $(x^0, y^0, s^0) := \gamma_0\rho(e, 0, e)$. Pro $k = 0, 1, 2 \ldots$ proveďme

Krok 1: Je-li

$$x^T s \leq \epsilon \quad \& \quad \|Ax^k - b\| \leq \epsilon_P \quad \& \quad \|A^T y + s^k - c\| \leq \epsilon_D \tag{Q1}$$

nebo

$$\|(x^k, s^k)\|_1 \geq \frac{1 + \gamma_0}{\gamma_0^2 r^k \rho}(x^k)^T s^k \ \text{ a } \ r^k > 0 \tag{Q2}$$

pak **STOP**, jinak

Krok 2: Položme $\tau := \beta_1 \frac{x^T s}{n}$ a řešením (C.5) spočtěme prediktorový směr $(\Delta x^p, \Delta y^p, \Delta s^p)$.

Krok 3: Buď $\bar{\alpha}^k$ největší z $\tilde{\alpha} \leq 1$ takových , že vztahy

$$\begin{aligned}
(x^k, y^k, s^k) + \alpha(\Delta x^p, \Delta y^p, \Delta s^p) &\in \mathcal{N}_2(\gamma_1), && \text{(C.13)}\\
(x^k + \alpha\Delta x^p)^T(s^k + \alpha\Delta s^p) &\leq (1 - \alpha(1 - \beta_2))(x^k)^T s^k, && \text{(C.14)}\\
(x^k + \alpha\Delta x^p)^T(s^k + \alpha\Delta s^p) &\geq (1 - \alpha) r^k (x^1)^T s^1 && \text{(C.15)}
\end{aligned}$$

platí pro všechna $\alpha \in \langle 0, \tilde{\alpha}\rangle$.

Krok 4. Položme

$$(\hat{x}^k, \hat{y}^k, \hat{s}^k) := (x^k, y^k, s^k) + \bar{\alpha}_k(\Delta x^p, \Delta y^p, \Delta s^p)$$

a

$$r^{k+1} := (1 - \bar{\alpha}_k) r^k$$

Krok 5. Definujme $\hat{\mu}^k := \frac{(\hat{x}^k)^T \hat{s}^k}{n}$ a vyřešme soustavu pro korektorový směr

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ \hat{S}^k & 0 & \hat{X}^k \end{bmatrix} \begin{bmatrix} \Delta x^c \\ \Delta y^c \\ \Delta s^c \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \hat{\mu}^k e - \hat{X}^k \hat{S}^k e \end{bmatrix}. \tag{C.16}$$

Krok 6. Nová iterace má potom tvar

$$(x^{k+1}, y^{k+1}, s^{k+1}) = (\hat{x}^k, \hat{y}^k, \hat{s}^k) + (\Delta x^c, \Delta y^c, \Delta s^c)$$

Počáteční přiblížení volíme jako

$$(x^0, y^0, s^0) := \gamma_0\rho(e, 0, e) \tag{C.17}$$

kde $\rho \geq \rho_0$ definované v (C.10) Všimněme si, že stejná volba počátečního přiblížení měla své teoretické opodstatnění, uvedené v kapitole 4.

Uvědomme si, že ze vztahů (C.5), (C.15) a (C.16) vyplývá

$$(Ax^k - b, A^T y^k + s^k - c) = r^k(Ax^1 - b, A^T y^1 + s^1 - c) \tag{C.18}$$

a

$$(x^k)^T s^k \geq r^k (x^1)^T s^1 \tag{C.19}$$

a podívejme se nyní, co znamenají podmínky (Q1) a (Q2). Je zřejmé, že je-li v k-té iteraci splněna podmínka (Q1), je $(x^k, y^k, s^k) \in \Omega_\epsilon$ a je tudíž dostatečně přesným přiblížením optimálního řešení (x^*, y^*, s^*). Podmínku (Q2) popisuje následující lemma.

Lemma 17 *Jestliže algoritmus skončí splněním podmínky (Q2), neexistuje optimální řešení x^* úlohy (P) a (y^*, s^*) úlohy (D), pro nějž*

$$\|(x^*, s^*)\|_\infty \leq \rho.$$

Důkaz: Předpokládejme, že jsme nalezli optimální řešení x^* úlohy (P) a (y^*, s^*) úlohy (D) taková, pro něž je $\|(x^*, s^*)\|_\infty \leq \rho$. Podle (C.18) potom platí

$$A(r^k x_1 + (1 - r^k)x^* - x^k) = 0,$$
$$A^T(r^k y_1 + (1 - r^k)y^* - y^k) + (r^k s_1 + (1 - r^k)s^* - s^k) = 0.$$

A tedy

$$(r^k x_1 + (1 - r^k)x^* - x^k)^T (r^k s_1 + (1 - r^k)s^* - s^k) = 0$$

z čehož plyne

$$\begin{aligned} & (r^k x_1 + (1 - r^k)x^*)^T s^k + (r^k s_1 + (1 - r^k)s^*)^T x^k \\ = \; & (r^k x_1 + (1 - r^k)x^*)^T (r^k s_1 + (1 - r^k)s^*) + (x^k)^T s^k \end{aligned}$$

Užijeme-li rovnosti $x^1 = s^1 = \gamma_0 \rho e,\ x^* \leq \rho e,\ s^* \leq \rho e$ a $x_i^* s_i^* = 0$ pro každé i, získáme vztah

$$\begin{aligned} r^k(\gamma_0\rho)\|(x^k, s^k)\|_1 &= r^k((s^1)^T x^k + (x^1)^T s^k) \\ &\leq (r^k x^1 + (1 - r^k)x^*)^T s^k + (r^k s_1 + (1 - r^k)s^*)^T x^k \\ &= (r^k x_1 + (1 - r^k)x^*)^T (r^k s_1 + (1 - r^k)s^*) + (x^k)^T s^k \\ &\leq n r^k \gamma_0 \rho^2 + (x^k)^T s^k. \end{aligned}$$

Podle (C.19) je $(x^k)^T s^k \geq r^k (x^1)^T s^1 = n r^k \gamma_0^2 \rho^2$. Tudíž

$$r^k \gamma_0 \rho \|(x^k, s^k)\|_1 \leq (1 + 1/\gamma_0)(x^k)^T s^k. \ \square$$

Z následujícího lemmatu a z faktu $(x^0, y^0, s^0) \in \mathcal{N}_2(\frac{1}{4})$ vyplývá, že pro každé k je $(x^k, y^k, s^k) \in \mathcal{N}_2(\frac{1}{4})$.

Lemma 18 *Je-li* $(\hat{x}^k, \hat{y}^k, \hat{s}^k) \in \mathcal{N}_2(2\gamma_1)$, kde $\gamma_1 = \frac{1}{4}$ *a* $(\Delta x^c, \Delta y^c, \Delta s^c)$ *je řešením (C.16), pak*

$$(\hat{x}^k, \hat{y}^k, \hat{s}^k) + (\Delta x^c, \Delta y^c, \Delta s^c) \in \mathcal{N}_2(\gamma_1)$$

Uveďme ještě odhad délky kroku pro Algoritmus A.

Lemma 19 *Předpokládejme, že v k-té iteraci platí*

$$\|\Delta X^p \Delta s^p - ((\Delta x^p)^T \Delta s^p/n)e\| \le \eta \quad \text{a} \quad |(\Delta x^p)^T \Delta s^p| \le \eta \tag{C.20}$$

Je-li $(x^k, y^k, s^k) \in \mathcal{N}_2(\gamma_1)$, *potom* $\alpha^k \ge \alpha^{k*}$, *kde*

$$\alpha^{k*} = \min\left(\frac{1}{2}, \sqrt{\frac{\gamma_1 (x^k)^T s^k}{2n\eta}}, \frac{\beta_1 (x^k)^T s^k}{\eta}, \frac{(\beta_2 - \beta_1)(x^k)^T s^k}{\eta}\right)$$

Důkaz Ze vztahu (C.5) vyplývá

$$\begin{aligned} & (x^k + \alpha \Delta x^p)^T (s^k + \alpha \Delta s^p) \\ = \; & (x^k)^T s^k - \alpha((x^k)^T s^k - \beta_1 (x^k)^T s^k) + \alpha^2 (\Delta x^p)^T \Delta s^p \\ = \; & (1 - \alpha + \beta_1 \alpha)(x^k)^T s^k + \alpha^2 (\Delta x^p)^T \Delta s^p. \end{aligned}$$

Protože je $\alpha^{k*} \le \min\left(\frac{\beta_1 (x^k)^T s^k}{\eta}, \frac{(\beta_2 - \beta_1)(x^k)^T s^k}{\eta}\right)$ a $|(\Delta x^p)^T \Delta s^p| \le \eta$, je

$$\begin{aligned} & (1 - \alpha + \beta_1 \alpha)(x^k)^T s^k + \alpha^2 (\Delta x^p)^T \Delta s^p \\ = \; & (x^k)^T s^k + \alpha(\beta_1 (x^k)^T s^k + \alpha (\Delta x^p)^T \Delta s^p - (x^k)^T s^k) \\ \le \; & (x^k)^T s^k + \alpha(\beta_1 (x^k)^T s^k + (\beta_2 - \beta_1)(x^k)^T s^k - (x^k)^T s^k)) \\ \le \; & (1 - \alpha(1 - \beta_2))(x^k)^T s^k \end{aligned} \tag{C.21}$$

a navíc

$$(x^k + \alpha \Delta x^p)^T (s^k + \alpha \Delta s^p) \ge (1 - \alpha)(x^k)^T s^k \tag{C.22}$$

pro každé $0 \le \alpha \le \alpha^{k*}$. Z druhé nerovnosti plyne vztah (C.15). Zbývá tedy dokázat, že

$$\|(X^k + \alpha \Delta X^p)(s^k + \alpha \Delta s^p) - \mu(\alpha)e\| \le 2\gamma_1 \mu(\alpha)$$

pro každé $0 \le \alpha \le \alpha^{k*}$, kde

$$\begin{aligned} \mu(\alpha) &= (x^k + \alpha \Delta x^p)^T (s^k + \alpha \Delta s^p)/n \\ &= (1 - \alpha + \beta_1 \alpha)(x^k)^T s^k/n + \alpha^2 (\Delta x^p)^T \Delta s^p/n \end{aligned}$$

Je zřejmé, že

$$\begin{aligned}
& \|(X^k + \alpha\Delta X^p)(s^k + \alpha\Delta s^p) - \mu(\alpha)e\| \\
= \; & \|X^k s^k - \alpha(X^k s^k - \beta_1((x^k)^T s^k/n)e) + \alpha^2(\Delta X^p)\Delta s^p \\
& -((1-\alpha+\beta_1\alpha)(x^k)^T s^k/n + \alpha^2(\Delta x^p)^T\Delta s^p/n)e\| \\
\leq \; & (1-\alpha)\|X^k s^k - ((x^k)^T s^k/n)e\| + \alpha^2\|\Delta X^p \Delta s \; - ((\Delta x^p)\Delta s^p/n)e\| \\
\leq \; & (1-\alpha)\gamma_1(x^k)^T s^k/n + \frac{\gamma_1(x^k)^T s^k}{2n\eta}\eta \\
\leq \; & 2\gamma_1(1-\alpha)(x^k)^T s^k/n \\
\leq \; & 2\gamma_2\mu(\alpha),
\end{aligned}$$

kde poslední nerovnost vyplývá z (C.22)

Věta 23 *Jsou-li β_1, β_2 a γ_0 konstanty nezávislé na vstupních datech, skončí algoritmus po nejvýše $O(nL)$ iteracích, kde*

$$L = max\left(\log\left((x^0)^T s^0/\epsilon\right), \log\|Ax^0 - b\|/\epsilon_P), \log\|A^T y^0 + s \; - c\|/\epsilon_D)\right)$$

Důkaz této věty je obsažen v článku S.Mizuno [20].

C.2 Metody podle J.Miao

Další dvě metody jsem převzala z článku Jianming Miao [21]. V obou případech se opět jedná o metody prediktor–korektor, přičemž okolí centrální cesty je množina

$$\mathcal{N}(\gamma) := \{(x,s) : (x,s) > 0, \|XSe - \mu e\| \leq \gamma\mu\} \tag{C.23}$$

Algoritmus 1

Mějme dáno (x^0, y^0, s^0), přičemž $(x^0, s^0) \in \mathcal{N}(\frac{1}{4})$. Pro $k = 0, 1, 2, \ldots$ proveďme

Krok 1. Je -li $(x^k, y^k, s^k) \in \Omega_\epsilon$, pak **STOP**, jinak

Krok 2. Vyřešme následující soustavu pro prediktorový směr $(\Delta x^p, \Delta y^p, \Delta s^p)$

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} \Delta x^p \\ \Delta y^p \\ \Delta s^p \end{bmatrix} = - \begin{bmatrix} Ax^k - b \\ A^T y^k + s^k - c \\ X^k S^k e \end{bmatrix}. \tag{C.24}$$

Krok 3. Zvolme vhodnou délku kroku α_k a definujme

$$(\hat{x}^k, \hat{y}^k, \hat{s}^k) := (x^k, y^k, s^k) + \alpha_k(\Delta x^p, \Delta y^p, \Delta s^p)$$

Krok 4. Položme $\bar{\mu}^k := (1-\alpha_k)\mu^k$ a vyřešme soustavu pro korektorový výběr směru

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ \hat{S}^k & 0 & \hat{X}^k \end{bmatrix} \begin{bmatrix} \Delta x^c \\ \Delta y^c \\ \Delta s^c \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \bar{\mu}^k e - \hat{X}^k \hat{S}^k e \end{bmatrix}. \tag{C.25}$$

Krok 5. Nová iterace má potom tvar

$$(x^{k+1}, y^{k+1}, s^{k+1}) = (\hat{x}^k, \hat{y}^k, \hat{s}^k) + (\Delta x^c, \Delta y^c, \Delta s^c)$$

Je-li $(x^0, y^0, s^0) \in \mathcal{F}^0$, pak jsou $(x^k, y^k, s^k) \in \mathcal{F}^0$ pro všechna k a Algoritmus 1 je přípustným algoritmem vnitřního bodu.

Algoritmus 1 je modifikací přípustného algoritmu prediktor–korektor. Základní modifikace spočívá ve volbě parametru $\bar{\mu}^k$ namísto $\hat{\mu}^k$ v *Kroku 4*. Neplatí totiž, jako u přípustného algoritmu prediktor–korektor, $\hat{\mu}^k = (1-\alpha)\mu^k$. Důvodem je skutečnost, že součin $\Delta x^{pT} \Delta s^p$ není pro nepřípustné přiblížení (x^k, y^k, s^k) obecně nulový.

Stejně jako v první části definujme i zde

$$r_b := Ax - b, \qquad r_c := A^T y + s \ - c$$

$$\nu_0 := 1 \tag{C.26}$$

$$\nu_k := \prod_{j=0}^{k-1} (1-\alpha_j) \tag{C.27}$$

a označme opět

$$\begin{aligned} x(\alpha) &:= x^k + \alpha \Delta x^p \\ y(\alpha) &:= y^k + \alpha \Delta y^p \\ s(\alpha) &:= s^k + \alpha \Delta s^p \end{aligned} \tag{C.28}$$

Poznamenejme ještě, že pro korektorový výběr směru je

$$(\Delta x^c)^T \Delta s^c = 0.$$

Platí následující tvrzení

Lemma 20 *Buď* $\{(x^k, y^k, s^k)\}$ *posloupnost generovaná Algoritmem 1. Pak je*

$$r_b{}^{k+1} = (1-\alpha^k) r_b{}^k = \nu^{k+1} r_b{}^0$$

$$r_c{}^{k+1} = (1-\alpha^k) r_c{}^k = \nu^{k+1} r_c{}^0$$

a

$$(x^{k+1})^T s^{k+1} = (1-\alpha^k)(x^k)^T s^k = \nu^{k+1} (x^0)^T s^0.$$

Z lemmatu 20 je zřejmé, že je-li délka kroku $\alpha^k = 1$ pro nějaké k, je $(x^{k+1}, y^{k+1}, s^{k+1}) \in \Omega$. Tato extrémní situace v praxi ovšem většinou nenastane. Výběr délky kroku α^k je založen na následující úvaze. Pro danou konstantu $\beta > \gamma = \frac{1}{4}$ (v našem případě volíme $\beta = 2\gamma$) chceme zvolit takové α^k, aby bod $(\hat{x}^k, \hat{s}^k)$ splňoval vztah

$$\|\hat{X}^k \hat{S}^k e - \bar{\mu}^k e\| \leq \beta \bar{\mu}^k \tag{C.29}$$

a následující iterace splňovala

$$(x^{k+1}, s^{k+1}) \in \mathcal{N}(\gamma) \tag{C.30}$$

Lemma 21 *Nechť $\beta \in (0,1)$ je daná konstanta. Jestliže existuje kladné číslo $\bar{\alpha} < 1$ takové, že*

$$\| X(\alpha)S(\alpha)e - (1-\alpha)\mu^k e \| \leq \beta(1-\alpha)\mu^k \tag{C.31}$$

pro všechna $0 \leq \alpha \leq \bar{\alpha}$, pak $(x(\bar{\alpha}), s(\bar{\alpha})) > 0$.

Důkaz: Podle (C.31) je $X(\alpha)S(\alpha)e \geq (1-\beta)(1-\alpha)\mu^k e > 0$ pro všechna $0 \leq \alpha \leq \bar{\alpha}$. Z toho ihned plyne, že pro všechna taková α je $(x(\alpha), s(\alpha)) > 0$.□

Délku kroku α^k tedy volíme jako největší takové $\bar{\alpha} \leq 1$, že je splněna podmínka (C.31) pro $\beta = 2\gamma$. V Algoritmu 1 navíc vždy volíme $\gamma = \frac{1}{4}$.

Stejně jako u přípustné metody prediktor–korektor můžeme i zde nalézt odhad délky kroku takto:

Lemma 22 *Buď $\{(x^k, y^k, s^k)\}$ posloupnost generovaná Algoritmem 1. Pak*

$$\alpha^k \geq \alpha_1 \quad = \quad min\left(\frac{1}{2}, \sqrt{\frac{\mu^k}{8 \parallel \Delta X^p \Delta s^p \parallel}}\right) \tag{C.32}$$

Důkaz: Pro $\alpha \in \langle 0, 1\rangle$ platí podle (C.24) a (C.28)

$$\begin{aligned} & \|X(\alpha)s(\alpha) - (1-\alpha)\mu^k e\| \\ = \; & \|X^k s^k + \alpha(-X^k s^k) + \alpha^2 \Delta X^p \Delta s^p - (1-\alpha)\mu^k e\| \\ \leq \; & (1-\alpha)\|X^k s^k - \mu^k e\| + \alpha^2 \|\Delta X^p \Delta s^p\| \\ \leq \; & (1-\alpha)\frac{1}{4}\mu^k + \alpha^2 \|\Delta X^p \Delta s^p\| \end{aligned}$$

Pro všechna $0 \leq \alpha \leq \alpha_1$ je $\alpha \leq \frac{1}{2}$ a zároveň $8\alpha^2 \parallel \Delta X^p \Delta s^p \parallel \leq \mu^k$. Předchozí nerovnost potom dává

$$\|X(\alpha)s(\alpha) - (1-\alpha)\mu^k e\| \leq \frac{1}{4}(1-\alpha)\mu^k \left[1 + \frac{1}{2(1-\alpha)}\right] \leq \frac{1}{2}(1-\alpha)\mu^k \tag{C.33}$$

Pak tedy, podle definice délky kroku, platí $\alpha^k \geq \alpha_1$, což bylo dokázati. □

Věta 24 *Buď* $\{(x^k, y^k, s^k)\}$ *posloupnost generovaná Algoritmem 1 a buď množina* $\mathcal{F}$ *neprázdná.Potom konvergují posloupnosti* $\{(x^k)^T s^k\}$, $\{{r_b}^k\}$, $\{{r_c}^k\}$ *Q-lineárně k nule.*

Důkaz: Základní myšlenka důkazu je tato. Nejprve dokážeme, že existuje konstanta ϑ_1 (může obecně záviset na vstupních datech a hodnotě n), pro niž

$$\| (x^k, y^k, s^k) \| \leq \vartheta_1$$

(viz například [22]). Poté využijeme tohoto odhadu, a dokážeme existenci takové konstanty ϑ_2, že

$$\| \Delta X^p \Delta s^p \| \leq \vartheta_2 \big((x^k)^T s^k\big)^2$$

(viz například [23]). Z lemmatu 22 potom plyne

$$\begin{aligned} \alpha^k &\geq min\left(\frac{1}{2}, \sqrt{\frac{1}{8n\vartheta_2 (x^k)^T s^k}}\right) \\ &\geq min\left(\frac{1}{2}, \sqrt{\frac{1}{8n\vartheta_2 (x^0)^T s^0}}\right) := \alpha_0 > 0 \end{aligned} \tag{C.34}$$

Z tohoto odhadu a z lemmatu 20 plyne tvrzení věty 24. □

Následující lemma uvádí jiný odhad pro dolní hranici hodnoty α^k.

Lemma 23 *Buď* $\{(x^k, y^k, s^k)\}$ *posloupnost generovaná Algoritmem 1. Pak*

$$\alpha^k \geq \frac{2}{1 + \sqrt{1 + 16 \| \Delta X^p \Delta s^p / \mu^k \|}} \tag{C.35}$$

Důkaz: Z důkazu lemmatu 22 víme, že

$$\| X(\alpha)s(\alpha) - (1-\alpha)\mu^k e \| \leq \frac{1}{4} (1-\alpha)\ \mu^k + \alpha^2\ \| \Delta X^p \Delta s^p \|$$

Vztahem

$$\frac{1}{4}(1-\alpha)\mu^k + \alpha^2 \| \Delta X^p \Delta s^p \| \leq \frac{1}{2}(1-\alpha)\mu^k \tag{C.36}$$

zaručíme splnění podmínky (C.31) pro $\beta = \frac{1}{2}$. Je zřejmé, že (C.36) je splněn pro všechna α menší nebo rovna kladnému kořenu

$$\alpha^+ = \frac{2}{1 + \sqrt{1 + 16 \| \Delta X^p \Delta s^p / \mu^k \|}} \tag{C.37}$$

kvadratické rovnice, která vznikne z (C.36) nahrazením nerovnosti rovností. Délku kroku lze potom (pro každé k) volit větší nebo rovnu α^+. Důkaz je hotov. □

Na základě Lemmatu 23 lze dokázat lemma 24

Lemma 24 *Buď* $\{(x^k, y^k, s^k)\}$ *posloupnost generovaná Algoritmem 1. Pak existuje konstanta* $\vartheta_3 > 0$ *taková, že*

$$1 - \alpha^k \leq \vartheta_3 (x^k)^T s^k \tag{C.38}$$

a konečně na základě lemmatu 24 lze odvodit větu 25

Věta 25 *Buď* $\{(x^k, y^k, s^k)\}$ *posloupnost generovaná Algoritmem 1 a buď množina* $\mathcal{F}$ *neprázdná.Potom posloupnosti* $\{(x^k)^T s^k)\}, \{{r_b}^k\}, \{{r_c}^k\}$ *konvergují k nule Q-kvadraticky.*

Důkaz: Podle lemmatu 20 a lemmatu 24 víme, že

$$(x^{k+1})^T s^{k+1} \leq \vartheta_3 \left[(x^k)^T s^k\right]^2$$

Q-kvadratickou konvergenci pak získáme ze vztahů

$$\| \ {r_b}^{k+1} \ \| \leq \vartheta_3 \frac{\mu^0}{\|{r_b}^0\|} \|{r_b}^k\|^2 \text{ a } \ \| \ {r_c}^{k+1} \ \| \leq \vartheta_3 \frac{\mu^0}{\|{r_c}^0\|} \|{r_c}^k\|^2 . \square$$

Jak už jsme se zmínili v první části, je pro dosažení polynomiální složitosti nutné zvolit počáteční přiblížení speciálního tvaru. Buď tedy (x^0, y^0, s^0) takové počáteční přiblížení, pro nějž je

$$(x^0, s^0) \in \mathcal{N}(\frac{1}{4}), \ x^* \leq \rho x^0, \ s^* \leq \rho s^0 \tag{C.39}$$

pro nějaké $(x^*, y^*, s^*) \in \mathcal{F}$ a $\rho \geq 1$. Poněvadž opět $(x^*, y^*, s^*) \in \mathcal{F}$ předem neznáme, volíme (x^0, y^0, s^0) jako u algoritmu A, tj. $(x^0, y^0, s^0) := \rho(e, 0, e)$, pro ρ dostatečně velké.

Věta 26 *Buď* $\{(x^k, y^k, s^k)\}$ *posloupnost generovaná Algoritmem 1 s počátečním přiblížením* (x^0, y^0, s^0)*, definovaným na základě (C.39). Pak pro každé* ϵ *nalezne algoritmus 1 přibližné řešení* $(x^k, y^k, s^k) \in \Omega_\epsilon$ *v nejvýše* $O(nL)$ *iteracích, kde*

$$L = max \left(\log \left((x^0)^T s^0/\epsilon\right), \log \|Ax^0 - b\|/\epsilon), \log \|A^T y^0 + s \ - c\|/\epsilon)\right)$$

Algoritmus 2

Mějme dáno (x^0, y^0, s^0), přičemž $(x^0, s^0) \in \mathcal{N}(\frac{1}{4})$. Pro $k = 0, 1, 2, \ldots$ proveďme

Krok 1. Je -li $(x^k, y^k, s^k) \in \Omega_\epsilon$, pak **STOP**, jinak

Krok 2. Vyřešme následující soustavu pro prediktorový směr $(\Delta x^p, \Delta y^p, \Delta s^p)$

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S^k & 0 & X^k \end{bmatrix} \begin{bmatrix} \Delta x^p \\ \Delta y^p \\ \Delta s^p \end{bmatrix} = - \begin{bmatrix} Ax^k - b \\ A^T y^k + s^k - c \\ \mu^k e \end{bmatrix}, \tag{C.40}$$

kde $\mu = \frac{1}{n}(x^k)^T s^k$

Krok 3. Zvolme vhodnou délku kroku α^k a definujme

$$(\hat{x}^k, \hat{y}^k, \hat{s}^k) := (x^k, y^k, s^k) + \alpha^k(\Delta x^p, \Delta y^p, \Delta s^p)$$

Krok 4. Položme $\bar{\mu}^k := (1 - \alpha_k)\mu^k$ a vyřešme soustavu pro korektorový výběr směru

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ \hat{S}^k & 0 & \hat{X}^k \end{bmatrix} \begin{bmatrix} \Delta x^c \\ \Delta y^c \\ \Delta s^c \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \bar{\mu}^k e - \hat{X}^k \hat{S}^k e \end{bmatrix}. \tag{C.41}$$

Krok 5. Nová iterace má potom tvar

$$(x^{k+1}, y^{k+1}, s^{k+1}) = (\hat{x}^k, \hat{y}^k, \hat{s}^k) + (\Delta x^c, \Delta y^c, \Delta s^c)$$

Algoritmus 2 je obdobou Algoritmu 1; liší se pouze ve výběru prediktorového směru. Délka prediktorového kroku α^k je rovněž volena jako největší možné α, pro které platí vztah (C.31), kde $\beta = \frac{1}{2}$. Znamená to, že posloupnost iterací generovaná oběma algoritmy má podobné teoretické vlastnosti. Lemmata 20 a 21 platí pro Algoritmus 2 ve stejné podobě. Jako jsme v lemmatu 22 našli odhad dolní hranice délky kroku Algoritmu 1, můžeme obdobným způsobem nalézt tento odhad pro Algoritmus 2.

Lemma 25 *Buď* $\{(x^k, y^k, s^k)\}$ *posloupnost generovaná Algoritmem 2. Pak*

$$\alpha^k \geq \alpha_2 = min\left(\frac{1}{4}, \sqrt{\frac{\mu^k}{8 \parallel \Delta X^p \Delta s^p \parallel}}\right) \tag{C.42}$$

Důkaz: Pro $\alpha \in \langle 0, 1\rangle$ je

$$\begin{aligned} \|X(\alpha)s(\alpha) - (1-\alpha)\mu^k e\| &= \|X^k s^k - \alpha\mu^k e + \alpha^2 \Delta X^p \Delta s^p - (1-\alpha)\mu^k e\| \\ &\leq \|X^k s^k - \mu^k e\| \ + \ \alpha^2 \|\Delta X^p \Delta s^p\| \\ &\leq \frac{1}{4}\mu^k + \alpha^2 \|\Delta X^p \Delta s^p\| \end{aligned}$$

Pro všechna $0 \leq \alpha \leq \alpha_2$ je $\alpha \leq \frac{1}{4}$ a zároveň $8\alpha^2 \|\Delta X^p \Delta s^p\| \leq \mu^k$. Předchozí nerovnost potom dává

$$\|X(\alpha)s(\alpha) - (1-\alpha)\mu^k e\| \leq \frac{1}{4}\mu^k + \frac{1}{8}\mu^k = \frac{3}{8}\mu^k \leq \frac{1}{2}(1-\alpha)\mu^k. \square \tag{C.43}$$

Podobně jako u Algoritmu 1, můžeme i zde nalézt dolní odhad pro délku kroku i jiným způsobem. Zaměřme se na nerovnost

$$\frac{1}{4}\mu^k + \alpha^2 \|\Delta X^p \Delta s^p\| \leq \frac{1}{2}(1-\alpha)\mu^k.$$

Je zřejmé, že tato nerovnost je splněna pro každé α menší než kladný kořen kvadratické rovnice

$$\frac{1}{4}\mu^k + \alpha^2 \|\Delta X^p \Delta s^p\| = \frac{1}{2}(1-\alpha)\mu^k.$$

který má tvar

$$\alpha^+ = \frac{1}{1 + \sqrt{1 + 4 \parallel \Delta X^p \Delta s^p / \mu^k \parallel}} \tag{C.44}$$

Zvolíme-li opět počáteční přiblížení podle (C.39), získáme polynomiální složitost Algoritmu 2.

Věta 27 *Buď* $\{(x^k, y^k, s^k)\}$ *posloupnost generovaná Algoritmem 2 s počátečním přiblížením* (x^0, y^0, s^0)*, definovaným na základě (C.39). Pak pro každé* ϵ *nalezne algoritmus 2 přibližné řešení* $(x^k, y^k, s^k) \in \Omega_\epsilon$ *v nejvýše* $O(nL)$ *iteracích, kde*

$$L = max\left(\log\left((x^0)^T s^0/\epsilon\right), \log \|Ax^0 - b\|/\epsilon), \log \|A^T y^0 + s \ - c\|/\epsilon)\right)$$

C.3 Programová realizace a závěry

V předchozích odstavcích jsem, na základě jmenovaných článků, teoreticky popsala tři algoritmy. V této části bych chtěla uvést některé aspekty jejich praktické realizace.

Nejdůležitějším programem pro řešení problému lineárního programování je program ULLPI1.I, který na základě hodnoty parametru MLP zvolí jeden ze tří základních algoritmů popsaných v odstavcích C.1 a C.2. Pro hodnotu MLP=1 řeší úlohu (P) a (D) Algoritmem 1, pro hodnotu MLP=2 Algoritmem 2 a konečně pro hodnotu MLP=3 Algoritmem A. Pro každý z nich tvoří podstatnou část vlastního numerického výpočtu vyřešení soustav lineárních rovnic pro prediktorový směr $(\Delta x^p, \Delta y^p, \Delta s^p)$ resp. korektorový směr $(\Delta x^c, \Delta y^c, \Delta s^c)$. Kromě hodnoty parametru MLP (která odpovídá volbě základního algoritmu) lze proto navíc volit tři různé hodnoty parametru NUMBER, na základě kterých vybírá ULLPI1.I vhodnou metodu pro tuto úlohu. ULLPI1.I tak ve skutečnosti zahrnuje devět různých způsobů řešení základního problému.

Pro vyřešení lineárních soustav jsem použila již hotové programy UDSLL1.I (NUMBER = 1), UDSLL2.I (NUMBER = 2) a UDSLL3.I (NUMBER = 3).Výpisy programů jsou obsaženy v části C.4, stejně jako výpis ULLPI1.I. Programy jsou součástí knihovny systému UFO.

Ať zvolíme základní algoritmus jakkoli, upravíme matici

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \tag{C.45}$$

způsobem uvedeným v příloze B na symetrický tvar (B.9) a v každé iteraci pak řešíme soustavy:

Algoritmus A

$$\begin{bmatrix} -D^{-2} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x^p \\ \Delta y^p \end{bmatrix} = -\begin{bmatrix} A^T y - c + \tau X^{-1} e \\ Ax - b \end{bmatrix} \tag{C.46}$$

$$\Delta s^p = -D^{-2}\Delta x^p - s + \tau X^{-1} e \tag{C.47}$$

$$\begin{bmatrix} -\hat{D}^{-2} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x^c \\ \Delta y^c \end{bmatrix} = -\begin{bmatrix} \hat{\mu}\hat{X}^{-1} e - \hat{s} \\ 0 \end{bmatrix} \tag{C.48}$$

$$\Delta s^c = -\hat{D}^{-2}\Delta x^c - \hat{s} + \hat{\mu}\hat{X}^{-1} e \tag{C.49}$$

Algoritmus 1

$$\begin{bmatrix} -D^{-2} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x^p \\ \Delta y^p \end{bmatrix} = -\begin{bmatrix} A^T y - c \\ Ax - b \end{bmatrix} \tag{C.50}$$

$$\Delta s^p = -D^{-2}\Delta x^p - s \tag{C.51}$$

$$\begin{bmatrix} -\hat{D}^{-2} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x^c \\ \Delta y^c \end{bmatrix} = -\begin{bmatrix} \bar{\mu}\hat{X}^{-1}e - \hat{s} \\ 0 \end{bmatrix} \tag{C.52}$$

$$\Delta s^c = -\hat{D}^{-2}\Delta x^c - s + \bar{\mu}\hat{X}^{-1}e \tag{C.53}$$

Algoritmus 2

$$\begin{bmatrix} -D^{-2} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x^p \\ \Delta y^p \end{bmatrix} = -\begin{bmatrix} A^T y + s - c + \mu X^{-1}e \\ Ax - b \end{bmatrix} \tag{C.54}$$

$$\Delta s^p = -D^{-2}\Delta x^p - s + \tau X^{-1}e \tag{C.55}$$

$$\begin{bmatrix} -\hat{D}^{-2} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x^c \\ \Delta y^c \end{bmatrix} = -\begin{bmatrix} \bar{\mu}\hat{X}^{-1}e - \hat{s} \\ 0 \end{bmatrix} \tag{C.56}$$

$$\Delta s^c = -\hat{D}^{-2}\Delta x^c - \hat{s} + \bar{\mu}\hat{X}^{-1}e \tag{C.57}$$

Popišme nyní jednotlivé metody pro řešení soustavy

$$\begin{bmatrix} -D^{-2} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = -\begin{bmatrix} \varphi \\ \psi \end{bmatrix}, \tag{C.58}$$

kde $[\varphi,\ \psi]$ je příslušný vektor pravé strany. Vztah pro Δs na tomto místě neuvažujme. Poněvadž závisí na konkrétní volbě základního algoritmu, nelze zobecnit a je tak pro každý algoritmus počítán zvlášť podle příslušných vzorců z hodnot Δx a Δy.

Program UDSLL1.UFO řeší soustavu (C.45) přímo. Převádí (C.58) na tvar

$$\begin{aligned} AD^2A^T\Delta y &= -\psi - AD^2\varphi, \\ \Delta x &= D^2A^T\Delta y + D^2\varphi, \end{aligned}$$

který jsme v odstavci B.2 nazvali normal equations. Z tohoto vyjádření lze pak vektor Δy vypočíst jako

$$\Delta y = -(AD^2A^T)^{-1}(\psi + AD^2\varphi). \tag{C.59}$$

Matice AD^2A^T je symetrická a pozitivně definitní (neboť matice D^2 je pozitivně definitní). Pro výpočet inverzní matice lze proto použít Choleského rozkladu (Gill-Murrayova????) pro řídké matice a rozložit AD^2A^T na součin LML^T, kde L je dolní

trojúhelníková a M diagonální matice. Tato metoda je vhodná pro takové (řídké) matice, pro něž je i AD^2A^T řídká a konverguje pro ně podstatně rychleji než další dvě metody. Ty jsou vhodné pro třídu matic, které požadavek řídkosti AD^2A^T nesplňují.

Program UDSLL2.I řeší soustavu (C.58) Bunch-Parlettovou metodou. Jedná se opět o přímou metodu, která ovšem počítá vektor $(\Delta x, \Delta y)$ z původní soustavy

$$\begin{bmatrix} -D^{-2} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = -\begin{bmatrix} \varphi \\ \psi \end{bmatrix}. \tag{C.60}$$

$$\tag{C.61}$$

Matici

$$\tilde{A} = \begin{bmatrix} -D^{-2} & A^T \\ A & 0 \end{bmatrix} \tag{C.62}$$

rozkládá způsobem $\tilde{A} = LNL^T$, kde matice N je blokově diagonální, přičemž bloky na diagonále jsou čtvercové matice řádu 1, případně 2 a L je dolní trojúhelníková matice, a řeší pak soustavu

$$LNL^T \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = -\begin{bmatrix} \varphi \\ \psi \end{bmatrix}. \tag{C.63}$$

$$\tag{C.64}$$

Pivotace je volena tak, aby byla zachována stabilita metody. Metoda je opět vhodná pro řídké matice. Pro plné matice selhává z důvodu velkého počtu kroků potřebných k faktorizaci.(???)

Konečně posledním z programů na řešení soustavy (C.45) je program UDSLL3.I. Tento program řeší soustavu

$$(AD^2A^T)\Delta y = -\psi - AD^2\varphi$$

iteračně metodou sdružených gradientů.Pro metodu sdružených gradientů je v každé iteraci nutné znát vektor $w = (AD^2A^T)\ v$, který je zde počítán po částech, tj.

$$\begin{aligned} w_1 &:= A^T v, \\ w_2 &:= D^2 w_1, \\ w &:= A\ w_2. \end{aligned}$$

Matice AD^2A^T se tak nikdy nevyčísluje, čímž se metoda zjednoduší a zrychlí. UDSLL3.I je možné použít pro jakékoliv (řídké i neřídké) matice, ale pro řídké matice konverguje pomalu.

Srovnání rychlosti konvergence a počtu iterací pro soubor dvanácti testovacích příkladů při různé volbě parametrů MLP a NUMBER uvádím na konci tohoto odstavce.

Nyní se zaměřme na rozbor algoritmů z odstavců C.1 a C.2. Algoritmus A, na rozdíl od zbývajících dvou algoritmů, vyžaduje volbu několika vstupních parametrů. Metoda konvergovala nejrychleji pro hodnoty $\beta_1 := 0.25,\ \beta_2 := 0.5$. (Tato volba parametrů je

mimo jiné uvedena také v T.Terlaky [13], odstavec 5.6.) Pro hodnoty parametrů $\beta_1,\ \beta_2$ blízké těmto se však počet iterací o mnoho nelišil. Mnohem větší vliv na rychlost konvergence algoritmů měla však volba délky kroku α^k; v předchozích odstavcích byly uvedené teoretické odhady hodnot α^k pro všechny tři algoritmy. Pro Algoritmus A jsem definovala α^k jako α^{k*} z lemmatu 19, pro Algoritmus 1 resp. Algoritmus 2 jsem za délku kroku volila hodnoty α^+ definovanou v (C.44) resp. (**??**). Ukázalo se však, že u Algoritmu A lze délku kroku ve skutečnosti volit 1.99 krát větší než je teoretický odhad (19); u Algoritmu 2 lze tuto hodnotu volit dokonce 1.999 krát větší než odhad (**??**). Počet iterací potřebných k nalezení dostatečně přesné aproximace optima se při těchto volbách délek kroku podstatně sníží.

Teoreticky odvozené počáteční přiblížení pro Algoritmus A resp. Algoritmus 1 a Algoritmus 2 má tvar $\gamma_0\rho(e,0,e)$ resp. $\rho(e,0,e)$.Zvolme tedy hodnotu $\gamma_0 := 1$ a zaměřme se pouze na volbu parametru ρ. Teoreticky odvozenou hodnotu ρ_0, definovanou vztahem (C.10) je v praxi obtížné nalézt a poněvadž hodnotu $\rho \geq \|(x^*, s^*)\|_\infty$ jsme obvykle schopni zjistit až po skončení algoritmu, volila jsem pro všechny tři algoritmy $\rho := 0,5.10^2$. (Algoritmy konvergovaly nejrychleji pro $\rho = \|(x^*, s^*)\|_\infty$.)

Konečně hodnoty $\epsilon,\ \epsilon_P,\ \epsilon_D$ pro Algoritmus A resp. ϵ pro Algoritmy 1 a 2 jsem volila jako $\epsilon = \epsilon_P = \epsilon_D = \epsilon = 10^{-6}$. U Algoritmu 1 a Algoritmu 2 se ale v posledním kroku hodnota α^k téměř rovnala jedné a proto jsem přibližné řešení $(x^\epsilon, y^\epsilon, s^\epsilon)$ získala s mnohem vyšší přesností.

Kromě těchto tří existuje ještě celá řada jiných metod vnitřního bodu. Ráda bych nejprve upozornila na článek Kojima & Megiddo & Mizuno [24], kde je hlubší teoretický rozbor Algoritmu 1. a další článek, který bych zde chtěla zmínit, je R.Tapia et al. [26] kde je pro řešení rovnice F(x,y,s) = 0 použita zcela jiná modifikace klasické Newtonovy metody. Některé odlišné přístupy k metodám vnitřního bodu lze rovněž nalézt v knihách S.Wright [2], T.Terlaky [13], B.Jansen [19].

```
AIgoritmus 1


MLP=1
NUMBER=1

 1 NIC= 0 NIT= 32 NFV= 1 F= .562D+03 C= .106D-07
 2 NIC= 0 NIT= 40 NFV= 1 F= .984D+02 C= .262D-06
 3 NIC= 0 NIT= 19 NFV= 1 F= .510D+02 C= .132D-10
 4 NIC= 0 NIT= 37 NFV= 1 F= .200D+03 C= .304D-10
 5 NIC= 0 NIT= 37 NFV= 1 F= .420D+03 C= .207D-06
 6 NIC= 0 NIT= 26 NFV= 1 F= .337D+02 C= .891D-09
 7 NIC= 0 NIT= 32 NFV= 1 F= .562D+03 C= .106D-07
 8 NIC= 0 NIT= 43 NFV= 1 F= .529D+03 C= .533D-07
 9 NIC= 0 NIT= 37 NFV= 1 F= .667D+03 C= .228D-06
10 NIC= 0 NIT= 30 NFV= 1 F= .539D+03 C= .105D-07
11 NIC= 0 NIT= 35 NFV= 1 F= .399D+03 C= .131D-10
12 NIC= 0 NIT= 33 NFV= 1 F= .726D+03 C= .275D-06
TOTAL   NIT= 401   NFV=  12   NFG=  0   NDC=   0 * 0
        NCG=   0   NRS=   0   NAD=  0   NRM=   0
TIME= 0:00:02.31


MLP=1
NUMBER=2

 1 NIC= 0 NIT= 32 NFV= 1 F= .562D+03 C= .106D-07
 2 NIC= 0 NIT= 40 NFV= 1 F= .984D+02 C= .262D-06
 3 NIC= 0 NIT= 19 NFV= 1 F= .510D+02 C= .132D-10
 4 NIC= 0 NIT= 37 NFV= 1 F= .200D+03 C= .304D-10
 5 NIC= 0 NIT= 37 NFV= 1 F= .420D+03 C= .207D-06
 6 NIC= 0 NIT= 26 NFV= 1 F= .337D+02 C= .893D-09
 7 NIC= 0 NIT= 32 NFV= 1 F= .562D+03 C= .106D-07
 8 NIC= 0 NIT= 43 NFV= 1 F= .529D+03 C= .533D-07
 9 NIC= 0 NIT= 37 NFV= 1 F= .667D+03 C= .228D-06
10 NIC= 0 NIT= 30 NFV= 1 F= .539D+03 C= .105D-07
11 NIC= 0 NIT= 35 NFV= 1 F= .399D+03 C= .347D-11
12 NIC= 0 NIT= 33 NFV= 1 F= .726D+03 C= .275D-06
TOTAL   NIT= 401   NFV=  12   NFG=  0   NDC=   0 * 0
        NCG=   0   NRS=   0   NAD=  0   NRM=   0
TIME= 0:00:04.83
```

```
AIgoritmus 2


MLP=2
NUMBER=1

 1 NIC= 0 NIT= 18 NFV= 1 F= .562D+03 C= .814D-07
 2 NIC= 0 NIT= 22 NFV= 1 F= .984D+02 C= .261D-07
 3 NIC= 0 NIT= 12 NFV= 1 F= .510D+02 C= .151D-09
 4 NIC= 0 NIT= 20 NFV= 1 F= .200D+03 C= .271D-07
 5 NIC= 0 NIT= 21 NFV= 1 F= .420D+03 C= .166D-09
 6 NIC= 0 NIT= 21 NFV= 1 F= .337D+02 C= .150D-08
 7 NIC= 0 NIT= 18 NFV= 1 F= .562D+03 C= .814D-07
 8 NIC= 0 NIT= 36 NFV= 1 F= .529D+03 C= .101D-06
 9 NIC= 0 NIT= 20 NFV= 1 F= .667D+03 C= .486D-07
10 NIC= 0 NIT= 17 NFV= 1 F= .539D+03 C= .460D-08
11 NIC= 0 NIT= 19 NFV= 1 F= .399D+03 C= .176D-07
12 NIC= 0 NIT= 18 NFV= 1 F= .726D+03 C= .930D-07
TOTAL   NIT= 242   NFV=  12   NFG=  0   NDC=   0 * 0
        NCG=   0   NRS=   0   NAD=  0   NRM=   0
TIME= 0:00:01.37


MLP=2
NUMBER=2

 1 NIC= 0 NIT= 18 NFV= 1 F= .562D+03 C= .816D-07
 2 NIC= 0 NIT= 22 NFV= 1 F= .984D+02 C= .261D-07
 3 NIC= 0 NIT= 12 NFV= 1 F= .510D+02 C= .151D-09
 4 NIC= 0 NIT= 20 NFV= 1 F= .200D+03 C= .271D-07
 5 NIC= 0 NIT= 21 NFV= 1 F= .420D+03 C= .167D-09
 6 NIC= 0 NIT= 15 NFV= 1 F= .337D+02 C= .114D-08
 7 NIC= 0 NIT= 18 NFV= 1 F= .562D+03 C= .816D-07
 8 NIC= 0 NIT= 23 NFV= 1 F= .529D+03 C= .182D-07
 9 NIC= 0 NIT= 20 NFV= 1 F= .667D+03 C= .486D-07
10 NIC= 0 NIT= 17 NFV= 1 F= .539D+03 C= .460D-08
11 NIC= 0 NIT= 19 NFV= 1 F= .399D+03 C= .176D-07
12 NIC= 0 NIT= 18 NFV= 1 F= .726D+03 C= .930D-07
TOTAL   NIT= 223   NFV=  12   NFG=  0   NDC= 446 * 0
        NCG=   0   NRS=   0   NAD=  0   NRM=   0
TIME= 0:00:02.75
```

```
Algoritmus 2


MLP=2
NUMBER=1

 1 NIC= 0 NIT= 18 NFV= 1 F= .562D+03 C= .814D-07
 2 NIC= 0 NIT= 22 NFV= 1 F= .984D+02 C= .261D-07
 3 NIC= 0 NIT= 12 NFV= 1 F= .510D+02 C= .151D-09
 4 NIC= 0 NIT= 20 NFV= 1 F= .200D+03 C= .271D-07
 5 NIC= 0 NIT= 21 NFV= 1 F= .420D+03 C= .166D-09
 6 NIC= 0 NIT= 21 NFV= 1 F= .337D+02 C= .15QD-08
 7 NIC= 0 NIT= 18 NFV= 1 F= .562D+03 C= .814D-07
 8 NIC= 0 NIT= 36 NFV= 1 F= .529D+03 C= .101D-06
 9 NIC= 0 NIT= 20 NFV= 1 F= .667D+03 C= .486D-07
10 NIC= 0 NIT= 17 NFV= 1 F= .539D+03 C= .460D-08
11 NIC= 0 NIT= 19 NFV= 1 F= .399D+03 C= .176D-07
12 NIC= 0 NIT= 18 NFV= 1 F= .726D+03 C= .930D-07
TOTAL   NIT= 242   NFV=  12   NFG=  0   NDC=   0 * 0
        NCG=   0   NRS=   0   NAD=  0   NRM=   0
TIME.= 0:00:01.37


MLP=2
NUMBER=2

 1 NIC= 0 NIT= 18 NFV= 1 F= .562D+03 C= .816D-07
 2 NIC= 0 NIT= 22 NFV= 1 F= .984D+02 C= .261D-07
 3 NIC= 0 NIT= 12 NFV= 1 F= .510D+02 C= .151D-09
 4 NIC= 0 NIT= 20 NFV= 1 F= .200D+03 C= .271D-07
 5 NIC= 0 NIT= 21 NFV= 1 F= .420D+03 C= .167D-09
 6 NIC= 0 NIT= 15 NFV= 1 F= .337D+02 C= .114D-08
 7 NIC= 0 NIT= 18 NFV= 1 F= .562D+03 C= .816D-07
 8 NIC= 0 NIT= 23 NFV= 1 F= .529D+03 C= .182D-07
 9 NIC= 0 NIT= 20 NFV= 1 F= .667D+03 C= .486D-07
10 NIC= 0 NIT= 17 NFV= 1 F= .539D+03 C= .460D-08
11 NIC= 0 NIT= 19 NFV= 1 F= .399D+03 C= .176D-07
12 NIC= 0 NIT= 18 NFV= 1 F= .726D+03 C= .930D-07
TOTAL   NIT= 223   NFV=  12   NFG=  0   NDC= 446 * 0
        NCG=   0   NRS=   0   NAD=  0   NRM=   0
TIME= 0:00:02.75
```

```
Algoritmus 2


MLP=3
NUMBER=1

 1 NIC= 0 NIT= 71 NFV= 1 F= .562D+03 C= .967D-12
 2 NIC= 0 NIT= 96 NFV= 1 F= .984D+02 C= .699D-11
 3 NIC= 0 NIT= 33 NFV= 1 F= .510D+02 C= .222D-15
 4 NIC= 0 NIT= 88 NFV= 1 F= .200D+03 C= .318D-11
 5 NIC= 0 NIT= 86 NFV= 1 F= .420D+03 C= .103D-11
 6 NIC= 0 NIT= 38 NFV= 1 F= .337D+02 C= .195D-05
 7 NIC= 0 NIT= 71 NFV= 1 F= .562D+03 C= .967D-12
 8 NIC= 0 NIT= 88 NFV= 1 F= .529D+03 C= .297D-10
 9 NIC= 0 NIT= 83 NFV= 1 F= .667D+03 C= .122D-11
10 NJC= 0 NIT= 82 NFV= 1 F= .539D+03 C= .187D-12
11 NIC= 0 NIT= 87 NFV= 1 F= .399D+03 C= .966D-11
12 NIC= 0 NIT= 76 NFV= 1 F= .726D+03 C= .350D-12
TOTAL   NIT= 899   NFV=  12   NFG=  0   NDC=   0 * 0
        NCG=   0   NRS=   0   NAD=  0   NRM=   0
TIME= 0:00:05.49


MLP=3
NUMBER=2

 1 NIC= 0 NIT= 71 NFV= 1 F= .562D+03 C= .453D-13
 2 NIC= 0 NIT= 96 NFV= 1 F= .984D+02 C= .610D-13
 3 NIC= 0 NIT= 33 NFV= 1 F= .510D+02 C= .111D-15
 4 NIC= 0 NIT= 88 NFV= 1 F= .200D+03 C= .637D-13
 5 NIC= 0 NIT= 86 NFV= 1 F= .420D+03 C= .207D-13
 6 NIC= 0 NIT= 57 NFV= 1 F= .337D+02 C= .383D-13
 7 NIC= 0 NIT= 71 NFV= 1 F= .562D+03 C= .453D-13
 8 NIC= 0 NIT= 88 NFV= 1 F= .529D+03 C= .223D-13
 9 NIC= 0 NIT= 83 NFV= 1 F= .667D+03 C= .376D-13
10 NIC= 0 NIT= 82 NFV= 1 F= .539D+03 C= .257D-13
11 NIC= 0 NIT= 87 NFV= 1 F= .399D+03 C= .395D-13
12 NIC= 0 NIT~ 76 NFV= 1 F= .726D+03 C= .262D-13
TOTAL   NIT= 918   NFV=  12   NFG=  0   NDC=1836 * 0
        NCG=   0   NRS=   0   NAD=  0   NRM=   0
TIME= 0:00:11.32
```

C.4 Výpisy programů

```
* SUBROUTINE ULLPI1                ALL SYSTEMS                98/12/01
C PORTABILITY : ALL SYSTEMS C 98/12/01 KO : ORIGINAL VERSION
*
* PURPOSE :
* INFEASIBLE PREDICTOR-CORRECTOR PRIMAL-DUAL INTERIOR POINT METHOD
* FOR LINEAR PROGRAMMING.
*
* PARAMETERS :
*  II  NF NUMBER OF VARIABLES.
*  II  NC NUMBER OF CONSTRAINTS.
*  II  MC  NUMBER OF ELEMENTS IN THE FIELD CG.
*  RI  C(NF)  VECTOR OF PRICES (GRADIENT OF THE LINEAR FUNCTION).
*  RI  CG(MC)  ELEMENTS OF THE CONSTRAINT JACOBIAN MATRIX.
*  II  ICG(NC+1)  POSITION OF THE FIRST ROWS ELEMENTS IN THE FIELD CG.
*  II  JCG(MC)  COLUMN INDICES OF ELEMENTS IN THE FIELD CG.
*  RI  B(NC)  RIGHT HAND SIDE VECTOR.
*  RO  X(NF)  VECTOR OF VARIABLES.
*  RA  Y(NC)  VECTOR OF LAGRANGE MULTIPLIERS.
*  RA  S(NF)  VECTOR OF SLACK VARIABLES.
*  RA  G(NF) GRADIENT OF THE LAGRANGIAN.
*  RA  CF(NC) CONSTRAINT VIOLATION VECTOR.
*  RA  D(NF)  DIAGONAL MATRIX.
*  RA  Z(2*NF+NC) DIRECTION VECTOR.
*  RA  VEC1(2*NF)  AUXILIARY VECTOR.
*  RA  VEC2(NF)  AUXILIARY VECTOR.
*
* COMMON DATA :
*  II  MMAX LENGTH OF THE ARRAYS JB,B.
*  RI  ETA0  MACHINE PRECISION.
*  RI  ETA2  TOLERANCE FOR POSITIVE DEFINITENESS.
*  TO  TDXX  TEXT INFORMATION ON THE DIRECTION VECTOR OBTAINED.
*
* SUBPROGRAMS USED :
*  S   UXSCMD  MULTIPLICATION OF A DENSE RECTANGULAR MATRIX STORED BY
*         COLUMNS BY A VECTOR WITH EVENTUAL ADDITION OF A SCALED VECTOR.
*  S   UXSRMD  MULTIPLICATION OF A DENSE RECTANGULAR MATRIX STORED BY
*         ROWS BY A VECTOR WITH EVENTUAL ADDITION OF A SCALED VECTOR.
*  S   UXVCOP  COPYING OF A VECTOR.
*  S   UXVDIR  VECTOR AUGMENTED BY THE SCALED VECTOR.
*  RF  UXVDOT  DOT PRODUCT OF TWO VECTORS.
*  S   UXVDIF  DIFFERENCE OF TWO VECTORS.
*  S   UXVMUL  DIAGONAL PREMULTIPLICATION OF A VECTOR.
*  S   UXVNEG  COPYING OF A VECTOR WITH CHANGE OF THE SIGN.
```

```
*  RF  UXVNOR  EUCLIDEAN NORM OF A VECTOR.
*  RF  UXVSAB  SUM OF ABSOLUTE VALUES OF VECTOR ELEMENTS.
*  S   UXVSET  INITIATION OF A VECTOR.
*  S   UXVSCL  SCALING OF A VECTOR.
*
* METHOD :
*
      SUBROUTINE ULLPI1(NF,NC,MC,C,CG,ICG,JCG,B,X,Y,G,CF,D,S,Z,VEC1,
     & VEC2,CMAX,GMAX,UMAX,RPF1,MLP,MRED,INEW)
      $INCLUDE('UMCOMM')
      INTEGER NF,NC,MC,ICG(NC+1),JCG(MC),MLP,MRED,INEW
      $FLOAT C(NF),CG(MC),B(NC),X(NF),Y(NC),G(NF),CF(NC),D(NF),S(NF),
     & Z(2*NF+NC),VEC1(2*NF),VEC2(NF),CMAX,GMAX,UMAX,RPF1
      $FLOAT POM,POM1,POM2,ALFA,BETA1,BETA2,GAMA1,ETA,RRO,RK
      $FLOAT INFEAS,UXVDOT,UXVNOR,UXVSAB
      INTEGER I
      SAVE BETA1,BETA2,GAMA1,RRO,RK
      IF (TSS(NS).NE.'ULPX') CALL UYPRO1('ULPX',1)
      GOTO (1,2,3) ISP(NS)
    1 CONTINUE
      CALL UOLLP1('ULLPI1')
      NRED=0
      RRO=0.5$P 2
      BETA1=0.25$P 0
      BETA2=0.5$P 0
      GAMA1=0.25$P 0
C
C     INITIAL APPROXIMATION
C
      RK=ONE
      CALL UXVSET(NF,RRO,X)
      CALL UXVSET(NF,RRO,S)
      CALL UXVSET(NC,ZERO,Y)
C
C     MAIN ITERATIVE CYCLE
C
   10 IF (NRED.GT.MRED) GO TO 74
      NRED=NRED+1
C
C     PREDICTOR
C
      RPF1=UXVDOT(NF,X,S)
      IF (MLP.EQ.3) RPF1=BETA1*RPF1
      RPF1=RPF1/$DBLE(NF)
      CALL UXVMUL(NF,X,S,D,-1)
C
```

```
C     RIGHT HAND SIDE PREPARATION FOR THE PREDICTOR STEP
C     CF = (CG)*X - B
C     G  = TRANS(CG)*Y - C + RPF1*INV(X)*E
C
      CALL UXSRMD(NF,NC,MC,CG,ICG,JCG,X,-ONE,B,CF)
      CALL UXSCMD(NF,NC,MC,CG,ICG,JCG,Y,-ONE,C,G)
      IF (MLP.EQ.1) THEN
      ELSE IF (MLP.EQ.2) THEN
      DO 11 I=1,NF
      G(I)=G(I)+S(I)-RPF1/X(I)
   11 CONTINUE
      ELSE
      DO 12 I=1,NF
      G(I)=G(I)+(RPF1/X(I))
   12 CONTINUE
      ENDIF
      CALL UYSET0(1,2)
      RETURN
C
C     DEFINITION OF THE NEW VARIABLES (X,Y,S) AFTER THE PREDICTOR STEP
C
    2 IF (MLP.EQ.1) THEN
      DO 13 I=1,NF
      Z(NF+NC+I)=-S(I)*(ONE+Z(I)/X(I))
   13 CONTINUE
      ELSE IF (MLP.EQ.2) THEN
      DO 14 I=1,NF
      Z(NF+NC+I)=-(S(I)*Z(I)+RPF1)/X(I)
   14 CONTINUE
      ELSE
      CALL UXSCMD(NF,NC,MC,CG,ICG,JCG,Z(NF+1),-ONE,C,VEC1)
      CALL UXVNEG(NF,VEC1,VEC1)
      CALL UXSCMD(NF,NC,MC,CG,ICG,JCG,Y,ONE,S,VEC2)
      CALL UXVDIF(NF,VEC1,VEC2,Z(NC+NF+1))
      ENDIF
      CALL UXVMUL(NF,Z(NF+NC+1),Z,VEC1,1)
      IF (MLP.EQ.1) THEN
      POM=UXVNOR(NF,VEC1)/RPF1
      ALFA=TWO/(ONE+SQRT(ABS(ONE+FOUR**2*POM)))
      ELSE IF (MLP.EQ.2) THEN
      POM=FOUR*UXVNOR(NF,VEC1)
      ALFA=ONE/(ONE+SQRT(ABS(ONE+POM/RPF1)))
      ALFA=1.9991$P 0*ALFA
      ELSE
      POM=UXVSAB(NF,VEC1)
      DO 16 I=1,NF
```

```
      VEC1(I)=VEC1(I)-POM/$DBLE(NF)
   16 CONTINUE
      POM=ABS(POM)
      POM1=UXVNOR(NF,VEC1)
      ETA=MAX(POM,POM1)
      POM2=UXVDOT(NF,X,S)
      POM=(POM2*GAMA1)/(2*$DBLE(NF)*ETA)
      POM=SQRT(POM)
      POM1=(BETA1*POM2)/ETA
      POM2=(BETA2-BETA1)*POM2/ETA
      ALFA=MIN(HALF,POM,POM1,POM2)
      ALFA=1.99$P 0*ALFA
      ENDIF
      CALL UXVDIR(NF,ALFA,Z,X,X)
      CALL UXVDIR(NC,ALFA,Z(NF+1),Y,Y)
      CALL UXVDIR(NF,ALFA,Z(NF+NC+1),S,S)
      IF (MLP.EQ.3) RK=(1-ALFA)*RK
C
C     CORRECTOR
C
      IF (MLP.LE.2) THEN
      RPF1=(ONE-ALFA)*RPF1
      ELSE
      RPF1=UXVDOT(NF,X,S)/$DBLE(NF)
      ENDIF
      CALL UXVMUL(NF,X,S,D,-1)
C
C     RIGHT HAND SIDE PREPARATION FOR THE CORRECTOR STEP
C     CF = 0
C     G  = - S + RPF1*INV(X)*E
C
      CALL UXVSET(NC,ZERO,CF)
      DO 19 I=1,NF
      G(I)=RPF1/X(I)-S(I)
   19 CONTINUE
      CALL UYSET0(1,3)
      RETURN
    3 CALL UXSCMD(NF,NC,MC,CG,ICG,JCG,Z(NF+1),0.0D 0,S,VEC1)
      CALL UXVNEG(NF,VEC1,Z(NF+NC+1))
C
C     DEFINITION OF THE NEW VARIABLES (X,Y,S) AFTER THE CORRECTOR STEP
C
      CALL UXVDIR(NF,ONE,Z,X,X)
      CALL UXVDIR(NC,ONE,Z(NF+1),Y,Y)
      CALL UXVDIR(NF,ONE,Z(NF+NC+1),S,S)
C
```

```
C     TERMINATION CRITERIA
C
      GMAX=UXVDOT(NF,X,S)
      CALL UXSRMD(NF,NC,MC,CG,ICG,JCG,X,-1.0D 0,B,VEC1)
      CMAX=UXVNOR(NC,VEC1)
      CALL UXVDIF(NF,S,C,VEC2)
      CALL UXSCMD(NF,NC,MC,CG,ICG,JCG,Y, 1.0D 0,VEC2,VEC1)
      UMAX=UXVNOR(NF,VEC1)
      CALL UXVCOP(NF,X,VEC1)
      CALL UXVCOP(NF,S,VEC1(NF+1))
      CALL UOLLP3(NF,NC,X,Y,S,G,CF,CMAX,GMAX,UMAX,INEW)
      IF ((GMAX.LE.TOLG).AND.(CMAX.LE.TOLC).AND.(UMAX.LE.TOLC)) THEN
      TUXX='OPTIMUM '
      GOTO 75
      ENDIF
      IF (MLP.EQ.3) THEN
      INFEAS=2*GMAX/(RK*RRO)
      POM=UXVSAB(2*NF,VEC1)
      IF (POM.GT.INFEAS) THEN
      TXFU='INFEAS  '
      GOTO 75
      ENDIF
      ENDIF
      GO TO 10
   74 CONTINUE
   75 CALL UOLLP2
      CALL UYEPI1(1)
      RETURN
      END
```

```
* SUBROUTINE UDSLL1                ALL SYSTEMS                98/12/01
C PORTABILITY : ALL SYSTEMS C 98/12/01 TU : ORIGINAL VERSION
*
* PURPOSE :
* DETERMINATION OF THE DIRECTION VECTOR AS A SOLUTION OF THE SYSTEM OF
* NORMAL EQUATIONS USING SPARSE GILL-MURRAY FACTORIZATION WITH THE
* CONTROL OF POSITIVE DEFINITENESS.
*
* PARAMETERS :
*  II  NF NUMBER OF VARIABLES.
*  II  NC NUMBER OF CONSTRAINTS.
*  II  MC  NUMBER OF ELEMENTS IN THE FIELD CG.
*  II  MMAX LENGTH OF THE ARRAYS JB,B.
*  RI  CG(MC)  ELEMENTS OF THE CONSTRAINT JACOBIAN MATRIX.
*  II  ICG(NC+1)  POSITION OF THE FIRST ROWS ELEMENTS IN THE FIELD CG.
*  II  JCG(MC)  COLUMN INDICES OF ELEMENTS IN THE FIELD CG.
*  RU  B(MMAX)  NUMERICAL VALUES OF THE SPARSE MATRIX B.
*         OF THE TRIANGULAR FACTOR.
*  II  IB(NC+1) POINTER VECTOR OF THE SPARSE MATRIX A.
*  II  JB(MMAX) INDICES OF THE TRIANGULAR FACTOR OF THE HESSIAN MATRIX.
*  RI  D(NF)  DIAGONAL MATRIX.
*  RI  G(NF) GRADIENT OF THE OBJECTIVE FUNCTION.
*  RI  CF(NC) CONSTRAINT VECTOR.
*  RO  S(NF+NC) DIRECTION VECTOR.
*  II  PSL(NC+1) POINTER VECTOR OF THE SPARSE TRIANGULAR FACTOR OF THE
*         HESSIAN MATRIX.
*  II  PERM(NC) PERMUTATION VECTOR.
*  IA  WN11(NC+1) AUXILIARY VECTOR.
*  IA  WN12(NC+1) AUXILIARY VECTOR.
*  RI  ETA2  TOLERANCE FOR POSITIVE DEFINITENESS.
*  IU  INF  DECOMPOSITION INDICATOR. INF=0 IF THE MATRIX IS POSITIVE
*         DEFINITE.
*
* COMMON DATA :
*  IO  ITERD  TERMINATION INDICATOR. ITERD<0-BAD DECOMPOSITION.
*         ITERD=0-DESCENT DIRECTION. ITERD=1-NEWTON LIKE STEP.
*         ITERD=2-INEXACT NEWTON LIKE STEP. ITERD=3-BOUNDARY STEP.
*         ITERD=4-DIRECTION WITH THE NEGATIVE CURVATURE.
*         ITERD=5-MARQUARDT STEP.
*  IO  ITERM  TERMINATION INDICATOR. ITERM=0-SUCCESSFUL TERMINATION.
*  IU  IDECC  DECOMPOSITION INDICATOR. IDECC=0-NO DECOMPOSITION.
*         IDECC=1-GILL-MURRAY DECOMPOSITION. IDECC=2-BUNCH-PARLETT
*         DECOMPOSITION. IDECC=3-INVERSION.
*  IU  NDECC  NUMBER OF DECOMPOSITIONS.
*  TO  TDXX  TEXT INFORMATION ON THE DIRECTION VECTOR OBTAINED.
*
```

```
* SUBPROGRAMS USED :
*  S   UXSPCF  SPARSE NUMERICAL GILL-MURRAY FACTORIZATION OF A
*          SYMMETRIC MATRIX USING COMPACT FORM FOR FACTORS WITH
*          THE CONTROL OF POSITIVE DEFINITENESS.
*  S   UXSPCB  SPARSE BACK SUBSTITUTION WITH FACTORS IN COMPACT FORM.
*  RF  UXVDOT  DOT PRODUCT OF VECTORS.
*  S   UXVNEG  COPYING OF A VECTOR WITH A CHANGE OF THE SIGN.
*  S   UXVSFP  PERMUTATION OF A VECTOR.
*  S   UXVSBP  INVERSE PERMUTATION OF A VECTOR.
*  S   UOD1D1  PRINT OF ENTRY TO DIRECTION DETERMINATION.
*  S   UOD1D2  PRINT OF EXIT FROM DIRECTION DETERMINATION.
*  S   UOD1D5  PRINT OF INFORMATION DURING DIRECTION DETERMINATION.
*
      SUBROUTINE UDSLL1(NF,NC,MC,MH,MMAX,CG,ICG,JCG,B,IB,JB,D,G,CF,S,
     & PSL,PERM,WN11,WN12,ETA2,INF)
      $INCLUDE('UMCOMM')
      INTEGER NF,NC,MC,MMAX,ICG(NC+1),JCG(MC),IB(NC+1),JB(MMAX),
     & PSL(NC+1),PERM(NC),WN11(NC+1),WN12(NC+1),INF
      INTEGER L,I,MH
      $FLOAT CG(MC),B(MMAX),D(NF),G(NF),CF(NC),S(NF+NC),ETA2
      $FLOAT ALF,TAU
      CALL UOD1D1
      DO 111 I=1,NF
      S(I)=ONE/D(I)
  111 CONTINUE
      CALL UCENS1(NF,NC,MC,MMAX,B,IB,JB,CG,ICG,JCG,S)
      IDECC=0
      L=IB(NC+1)-1
      IF(IDECC.NE.1) THEN
      CALL UXSPCT(NC,L,MH,MMAX,B,JB,PSL,ITERM)
      IF (ITERM.NE.0) THEN
      TDXX='LACK SPC'
      GO TO 3
      ENDIF
C
C     GILL-MURRAY DECOMPOSITION
C
      ALF = ETA2
      CALL UXSPCF(NC,MMAX,B(L+1),PSL,JB(L+1),WN11,WN12,S,INF,ALF,TAU)
      NDECC=NDECC+1
      IDECC=1
      ENDIF
C
C     NEWTON LIKE STEP
C
      CALL UXVMUL(NF,D,G,S,-1)
```

```
      CALL UXSRMD(NF,NC,MC,CG,ICG,JCG,S,ONE,CF,S(NF+1))
      CALL UXVNEG(NC,S(NF+1),S(NF+1))
      CALL UXVSFP(NC,PERM,S(NF+1),S)
      CALL UXSPCB(NC,MMAX,B(L+1),PSL,JB(L+1),S(NF+1),0)
      CALL UXVSBP(NC,PERM,S(NF+1),S)
      CALL UXSCMD(NF,NC,MC,CG,ICG,JCG,S(NF+1),ONE,G,S)
      CALL UXVMUL(NF,D,S,S,-1)
      IF(INF.EQ.0) THEN
      ITERD = 1
      TDXX='G-M  POS'
      ELSEIF(INF.LT.0) THEN
      ITERD = 2
      TDXX='G-M  ZER'
      ELSE
      ITERD = 2
      TDXX='G-M  NEG'
      ENDIF
      CALL UOD1D5(ALF,TAU,INF)
    3 CALL UOD1D2(NF,G,S)
      RETURN
      END
```

```
* SUBROUTINE UDSLL2                ALL SYSTEMS                98/12/01
C PORTABILITY : ALL SYSTEMS C 98/12/01 LU : ORIGINAL VERSION
*
* PURPOSE :
* DETERMINATION OF THE DIRECTION VECTOR AS A SOLUTION OF THE INDEFINITE
* KARUSH-KUHN-TUCKER SYSTEM USING SPARSE BUNCH-PARLETT FACTORIZATION.
*
* PARAMETERS :
*  II  NF  NUMBER OF VARIABLES.
*  II  NC  NUMBER OF CONSTRAINTS.
*  II  MC  NUMBER OF ELEMENTS IN THE FIELD CG.
*  II  M  LENGTH OF THE ARRAY H.
*  II  MMAX  LENGTH OF THE ARRAY B.
*  RI  H(M)  SPARSE SYMMETRIC MATRIX WHICH IS USED FOR DETERMINATION
*         OF THE DIRECTION VECTOR.
*  II  IH(NF+1)  POINTERS OF THE DIAGONAL ELEMENTS OF H.
*  II  JH(M)  INDICES OF THE NONZERO ELEMENTS OF H.
*  RI  G(NF)  GRADIENT OF THE OBJECTIVE FUNCTION.
*  RO  S(NF+NC)  DIRECTION VECTOR.
*  RA  XO(NF+NC)  AUXILIARY VECTOR.
*  RA  GO(NF+NC)  AUXILIARY VECTOR.
*  RA  XS(NF+NC)  AUXILIARY VECTOR.
*  II  IX(MX)  VECTOR CONTAINING TYPES OF BOUNDS.
*
* COMMON DATA :
*  IO  ITERD  TERMINATION INDICATOR. ITERD<0-BAD DECOMPOSITION.
*         ITERD=0-DESCENT DIRECTION. ITERD=1-NEWTON LIKE STEP.
*         ITERD=2-INEXACT NEWTON LIKE STEP. ITERD=3-BOUNDARY STEP.
*         ITERD=4-DIRECTION WITH THE NEGATIVE CURVATURE.
*         ITERD=5-MARQUARDT STEP.
*  IO  ITERM  TERMINATION INDICATOR. ITERM=0-SUCCESSFUL TERMINATION.
*  IU  IDECC  DECOMPOSITION INDICATOR. IDECC=0-NO DECOMPOSITION.
*         IDECC=1-GILL-MURRAY DECOMPOSITION. IDECC=2-BUNCH-PARLETT
*         DECOMPOSITION. IDECC=3-INVERSION.
*  IU  NDECC  NUMBER OF DECOMPOSITIONS.
*  TO  TDXX  TEXT INFORMATION ON THE DIRECTION VECTOR OBTAINED.
*
* SUBPROGRAMS USED :
*  S   UNSTEP  DETERMINATION OF A SCALING FACTOR FOR THE BOUNDARY STEP.
*  S   UXSSMM  MATRIX-VECTOR PRODUCT.
*  RF  UXSSMQ  VALUE OF THE QUADRATIC FORM.
*  RF  UXUDEL  NORM OF THE AUGMENTED VECTOR.
*  S   UXUDIR  VECTOR AUGMENTED BY THE SCALED VECTOR.
*  RF  UXUDOT  DOT PRODUCT OF VECTORS.
*  S   UXVNEG  COPYING OF A VECTOR WITH CHANGE OF THE SIGN.
*  S   UXVSET  INITIATION OF A VECTOR.
```

```
*  S   UOD1D1  PRINT OF ENTRY TO DIRECTION DETERMINATION.
*  S   UOD1D2  PRINT OF EXIT FROM DIRECTION DETERMINATION.
*
      SUBROUTINE UDSLL2(NF,NC,MC,M,MMAX,H,IH,CG,JCG,KCG,B,IB,JB,KB,IW,
     & IW1,IW2,G,CF,S,XO,NAU,NUP,INF)
      $INCLUDE('UMCOMM')
      INTEGER NF,NC,MC,M,MMAX,IH(NF+1),JCG(MC),KCG(MC),IB(NF+NC+1),
     & JB(M+MC+NC),KB(M+MC+NC),IW(MMAX),IW1(2*(NF+NC)),IW2(3*(NF+NC)),
     & NAU,NUP,INF
      $FLOAT H(M),CG(MC),B(MMAX),G(NF),CF(NC),S(NF+NC),XO(NF+NC)
      INTEGER NN,MM
      CALL UOD1D1
      IDECF=0
      M=IH(NF+1)-1
      IF(IDECF.NE.7) THEN
      CALL UXSKIN(NF,M,H,IH,NC,MC,CG,JCG,KCG,B,IB)
      NN=NF+NC
      MM=M+MC+NC
C
C     BUNCH-PARLETT DECOMPOSITION
C
      CALL UXSIBF(NN,MM,KB,JB,B,MMAX,IW,MMAX,IW1,IW2,NAU,NUP,INF)
      IF (INF.LT.0) THEN
      TDXX='LACK SPC'
      ITERM=INF
      GO TO 3
      ENDIF
      NDECF=NDECF+1
      IDECF=7
      ENDIF
      CALL UXVNEG(NF,G,S)
      CALL UXVNEG(NC,CF,S(NF+1))
      CALL UXSIBB(NN,B,MMAX,IW,MMAX,XO,S,IW1,NAU,NUP)
C
C     NEWTON LIKE STEP
C
      ITERD=1
      TDXX = 'B-P STEP'
    3 CALL UOD1D2(NF,G,S)
      RETURN
      END
```

```
* SUBROUTINE UDSLL3                ALL SYSTEMS                97/12/01
C PORTABILITY : ALL SYSTEMS C 97/12/01 LU : ORIGINAL VERSION
*
* PURPOSE :
* DETERMINATION OF THE DIRECTION VECTOR AS A SOLUTION OF THE SYSTEM OF
* NORMAL EQUATIONS USING ITERATIVE CONJUGATE GRADIENT METHOD.
*
* PARAMETERS :
*  II  NF  NUMBER OF VARIABLES.
*  II  NC  NUMBER OF CONSTRAINTS.
*  II  MC  NUMBER OF ELEMENTS IN THE FIELD CG.
*  RI  CG(MC)  ELEMENTS OF THE CONSTRAINT JACOBIAN MATRIX.
*  II  ICG(NC+1)  POSITION OF THE FIRST ROWS ELEMENTS IN THE FIELD CG.
*  II  JCG(MC)  COLUMN INDICES OF ELEMENTS IN THE FIELD CG.
*  RU  B(MMAX)  NUMERICAL VALUES OF THE SPARSE MATRIX B.
*         OF THE TRIANGULAR FACTOR.
*  II  IB(NC+1) POINTER VECTOR OF THE SPARSE MATRIX A.
*  II  JB(MMAX) INDICES OF THE TRIANGULAR FACTOR OF THE HESSIAN MATRIX.
*  RI  D(NF)  DIAGONAL MATRIX.
*  RI  G(NF) GRADIENT OF THE OBJECTIVE FUNCTION.
*  RI  CF(NC) CONSTRAINT VECTOR.
*  RO  S(NF+NC) DIRECTION VECTOR.
*  II  PSL(NC+1) POINTER VECTOR OF THE SPARSE TRIANGULAR FACTOR OF THE
*         HESSIAN MATRIX.
*  II  PERM(NC) PERMUTATION VECTOR.
*  IA  WN11(NC+1) AUXILIARY VECTOR.
*  IA  WN12(NC+1) AUXILIARY VECTOR.
*  RI  ETA0  MACHINE PRECISION.
*
* COMMON DATA :
*  IO  ITERD  TERMINATION INDICATOR. ITERD<0-BAD DECOMPOSITION.
*         ITERD=0-DESCENT DIRECTION. ITERD=1-NEWTON LIKE STEP.
*         ITERD=2-INEXACT NEWTON LIKE STEP. ITERD=3-BOUNDARY STEP.
*         ITERD=4-DIRECTION WITH THE NEGATIVE CURVATURE.
*         ITERD=5-MARQUARDT STEP.
*  IU  IDECC  DECOMPOSITION INDICATOR. IDECC=0-NO DECOMPOSITION.
*         IDECC=1-GILL-MURRAY DECOMPOSITION. IDECC=2-BUNCH-PARLETT
*         DECOMPOSITION. IDECC=3-INVERSION.
*  TO  TDXX  TEXT INFORMATION ON THE DIRECTION VECTOR OBTAINED.
*
* SUBPROGRAMS USED :
*  S   UXSPCF  SPARSE NUMERICAL GILL-MURRAY FACTORIZATION OF A
*          SYMMETRIC MATRIX USING COMPACT FORM FOR FACTORS WITH
*          THE CONTROL OF POSITIVE DEFINITENESS.
*  S   UXSPCB  SPARSE BACK SUBSTITUTION WITH FACTORS IN COMPACT FORM.
*  RF  UXVDOT  DOT PRODUCT OF VECTORS.
```

```
*  S   UXVNEG  COPYING OF A VECTOR WITH A CHANGE OF THE SIGN.
*  S   UXVSFP  PERMUTATION OF A VECTOR.
*  S   UXVSBP  INVERSE PERMUTATION OF A VECTOR.
*  S   UOD1D1  PRINT OF ENTRY TO DIRECTION DETERMINATION.
*  S   UOD1D2  PRINT OF EXIT FROM DIRECTION DETERMINATION.
*  S   UOD1D5  PRINT OF INFORMATION DURING DIRECTION DETERMINATION.
*
      SUBROUTINE UDSLL3(NF,NC,MC,CG,ICG,JCG,D,G,CF,S,XO,GO,XS,GS,XP,GP,
     & ETA0,ETA9,SNORM,XDEL,MOS3)
      $INCLUDE('UMCOMM')
      INTEGER NF,NC,MC,ICG(NC+1),JCG(MC),MOS3
      INTEGER IMX,MMX
      $FLOAT CG(MC),D(NF),G(NF),CF(NC),S(NF+NC),XO(NC),GO(NC),
     & XS(NC),GS(NC),XP(NC),GP(NC),ETA0,ETA9,SNORM,XDEL
      $FLOAT  RHO,RHO1,RHO2,ALF,TEMP2,UXVDOT,UXVNOR,BTB,BTR,RMU,PAR
      CALL UOD1D1
      IDECC=0
C
C     NEWTON LIKE STEP
C
      CALL UXVMUL(NF,D,G,S,-1)
      CALL UXSRMD(NF,NC,MC,CG,ICG,JCG,S,ONE,CF,GS)
      CALL UXVNEG(NC,GS,GS)
      RHO1 = UXVNOR(NC,GS)
      PAR = SQRT(ETA0)
      CALL UXVSET(NC,ZERO,S(NF+1))
      CALL UXVSET(NC,ZERO,XP)
      CALL UXVCOP(NC,GS,XO)
      CALL UXVCOP(NC,GS,XS)
      RHO = UXVDOT(NC,XS,XS)
      MMX=2*NC
      DO 1 IMX=1,MMX
      CALL UXSCMM(NF,NC,MC,CG,ICG,JCG,XO,S)
      CALL UXVMUL(NF,D,S,S,-1)
      CALL UXSRMM(NF,NC,MC,CG,ICG,JCG,S,GO)
      ALF = UXVDOT(NC,XO,GO)
      IF (ALF.EQ.ZERO) THEN
      ITERD=-5
      TDXX='CG BREAK'
      CALL UOERR1('UDSLL3',5)
      GO TO 3
      ENDIF
C
C     CG STEP
C
      TDXX = 'CG  STEP'
```

```
      ALF = RHO/ALF
      CALL UXVDIR(NC,ALF,XO,XP,XP)
      CALL UXVDIR(NC,-ALF,GO,XS,XS)
      IF (MOS3.LE.0) THEN
      CALL UXVCOP(NC,XS,GS)
      CALL UXVCOP(NC,XP,S(NF+1))
      ELSE
      RMU=ONE/ETA9
      CALL UXVDIF(NC,GS,XS,GP)
      BTB=UXVDOT(NC,GP,GP)
      BTR=UXVDOT(NC,GP,XS)
      RMU=-BTR/MAX(BTB,RMU)
      CALL UXVDIR(NC,RMU,GP,XS,GS)
      CALL UXVDIF(NC,S(NF+1),XP,GP)
      CALL UXVDIR(NC,RMU,GP,XP,S(NF+1))
      ENDIF
      TEMP2 = UXVNOR(NC,GS)
      CALL UOD1D4(RHO1,TEMP2,PAR,SNORM,XDEL)
      IF (TEMP2.LE.PAR*RHO1) GO TO 2
      RHO2 = UXVDOT(NC,XS,XS)
      ALF = RHO2/RHO
      CALL UXVDIR(NC,ALF,XO,XS,XO)
      RHO = RHO2
    1 CONTINUE
      TDXX = 'IT LIMIT'
    2 ITERD=2
      CALL UXSCMD(NF,NC,MC,CG,ICG,JCG,S(NF+1),ONE,G,S)
      CALL UXVMUL(NF,D,S,S,-1)
    3 CALL UOD1D2(NF,G,S)
      RETURN
      END
```

Literatura

[1] N.Karmarkar: A new polynomial time algorithm for linear programming, Proceedings of the 16th Annual ACM Symposiumon the Theory of Computing, pp. 302-311

[2] S.J.Wright: Primal-dual interior point methods, SIAM, 1997

[3] D.Goldfarb & M.J. Todd : Linear Programming chapter II in G.L.Nemhauser et al., Eds., Handbooks in OR & MS, Vol.1, Elsevier Science Publishers B.V. (North-Holland) 1989

[4] A.V.Fiacco & G.P.McCormick: Nonlinear Programming: Sequential Unconstrained Minimization Techniques, John Wiley & sons, New York. Reprint: SIAM Classics in Applied Mathematics, vol.4, SIAM Publications, Philadelphia, Pennsylvania, 1990

[5] M.H.Wright:Interior methods for constrained optimization, Acta Numerica (1991), pp. 341-407

[6] J.Ji & Y.Ye: A complexity analysis for interior-point algorithms based on Karmarkar's potential function, SIAM J. OPTIMIZATION, Vol.4, No.3, pp.512-520, Aug.1994

[7] B.Jansen & C.Ross & T.Terlaky & J.P.Vial: Primal-dual Algorithms for Linear Programming Based on the Logarithmic Barrier Method, JOTA, Vol.83, No.1, Oct.1994

[8] D.F.Shanno & E.M.Simantiraki: Interior Point Methods for Linear and Nonlinear Programming,Rutgers Center of Operation Research, Rutgers University, NJ 08903

[9] R.D.C.Monteiro & I.Adler: Interior path–following primal-dual algorithms. Part I:Linear Programming, Mathematical Programming, 44 (1989), pp. 27-41

[10] S.Mizuno & M.Todd & Y.Ye: On adaptive step primal-dual interior-point algorithms for linear programming, Mathematics of Operations Research, 18 (1993), pp. 964-981

[11] G.Sonnevend & J.Stoer & G.Zhao: On the complexity of following the central path of linear programs by linear extrapolation, Methods of Operations Research, 62 (1989), pp. 19-31

[12] G.Sonnevend & J.Stoer & G.Zhao: On the complexity of following the central path of linear programs by linear extrapolation II, Mathematical Programming, 52 (1991), pp.527-553

[13] T.Terlaky(Ed.): Interior point methods of mathematical programming, Kluwer Academic Publishers,1996

[14] D.F.Shanno:Computational methods for linear programming, in E.Spedicato(ed.),Algorithms for Continuous Optimization, pp.383-413,Kluwer Academic Publishers,1994

[15] P.E.Gill & W.Murray & D.B.Ponceleon & M.A.Saunders: Primal-dual methods for linear programming, Mathematical Programming 70 (1995), pp.251-277

[16] E.R.Barnes: A variation on Karmarkar's algorithm for solving linear programming problems, Mathematical Programming 36 (1986), pp.174-182

[17] R.J.Vanderbei & M.S.Meketon & B.A.Freedman: A modification of Karmarkar's linear programming algorithm, Algorithmica 1(4), pp.395-407

[18] I.Adler & N.K.Karmarkar & M.G.C.Resende & G.Veiga: An implementation of Karmarkar's algorithm for linear programming, Mathematical Programming 44 (1986), pp.297-335

[19] B.Jansen: Interior point techniques in optimization, Complementarity, Sensitivity and Algorithms, Kluwer Academic Publishers,1997

[20] S.Mizuno: Polynomiality of infeasible-interior-point algorithms for linear programming, Mathematical Programming 67 (1994), pp. 109-119

[21] J.Miao: Two infeasible interior-point predictor-corrector algorithms for linear programming, SIAM J.Optimization, vol.6, No.3, pp. 587-599

[22] Y.Zhang: On the convergence of an infeasible interior-point algorithm for linear programming and other problems, SIAM J.Optim.,4 (1994), pp. 208-227

[23] Y.Zhang & D.Zhang: Superlinear Convergence of Infeasible Interior-Point Methods for Linear Programming, Tech.Report 92/15

[24] M.Kojima & N.Megiddo & S.Mizuno: A primal-dual infeasible-interior-point algorithm for linear programming, Mathematical Programming 61 (1993), pp. 263-280

[25] J.Stoer: Infeasible-interior-point methods for solving linear programs, in E.Spedicato(ed.),Algorithms for Continuous Optimization, pp.415-434,Kluwer Academic Publishers,1994

[26] R.Tapia & Y.Zhang & M.Saltzman & A.Weiser: The Mehrotra predictor–corrector method as a perturbed composite Newton method SIAM J.Optimization, vol.6, No.1, pp. 47-56, únor 1996.