UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

Metóda sečnicových nadrovín

DIPLOMOVÁ PRÁCA

Bratislava 2007 Miroslav Ďuriš

UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

Katedra aplikovanej matematiky a štatistiky

Metóda sečnicových nadrovín

DIPLOMOVÁ PRÁCA

Miroslav Ďuriš

Vedúci diplomovej práce: doc. RNDr. Milan Hamala, CSc.

Študijný odbor: Matematika

Špecializácia: Ekonomická a finančná matematika

Bratislava 2007

Čestne prehlasujem, že diplomovú som vypracoval samostatne s využitím teoretických vedomostí a s použitím uvedenej literatúry.

Bratislava, apríl 2007 Miroslav Ďuriš

Poďakovanie

Chcel by som poďakovať vedúcemu diplomovej práce doc. RNDr. Milanovi Hamalovi, CSc. za cenné rady a pripomienky, ktoré mi pomohli pri písaní tejto práce. Úprimne ďakujem aj svojim rodičom, ktorí mi umožnili študovať a za ich všestrannú podporu počas celého štúdia.

Abstrakt

Metóda sečnicových nadrovín je iteratívnou metódou riešenia úloh konvexného programovania s lineárnymi ohraničeniami. Každá iterácia pozostáva v riešení úlohy lineárneho programovania, pričom dve úlohy lineárneho programovania prislúchajúce po sebe nasledujúcim iteráciam sa líšia pridaním jedného ohraničenia. Cieľom práce je podrobne opísať, programovo realizovať a na experimentoch vyhodnotiť metódu sečnicových nadrovín pre rôzne účelové funkcie a rôzny počet premenných a ohraničení.

Kľúčové slová:

Metóda sečnicových nadrovín, Simplexová metóda, Cutting plane, Kelley

Obsah

Úvod

Metóda sečnicových nadrovín je iteratívnou metódou riešenia úloh konvexného programovania s lineárnymi ohraničeniami. Úlohy konvexného programovania sú úlohy, v ktorých hľadáme minimum alebo maximum konvexnej funkcie na množine prípustných riešení. Metóda sečnicových nadrovín rieši úlohy konvexného programovania, v ktorých množina prípustných riešení je tvorená lineárnymi ohraničeniami. Teda riešime úlohu

$$Min\{f(x)|\ Ax \leq b,\ x \geq 0\}$$

v ktorej hľadáme minimum konvexnej účelovej funkcie $f(x)$ so spojitými prvými parciálnymi deriváciami na množine prípustných riešení tvaru

$$K = \{x|\ Ax \leq b,\ x \geq 0\}$$

Prípustné riešenie $\hat{x} \in K$ pre ktoré platí, $f(\hat{x}) \leq f(x),\ \forall x \in K$ nazývame optimálnym riešením.

Metóda sečnicových nadrovín je iteratívna metóda, kde v každej iterácii sa rieši pomocná úloha lineárneho programovania. Pomocné úlohy lineárneho programovania dvoch po sebe nasledujúcich iterácii metódy sečnicových nadrovín sa odlišujú len pridaním jedného ohraničenia. Takže pomocná úloha lineárneho programovania, ktoú riešime v každej iterácii metódy sečnicových nadrovín sa postupne po iteráciach zväčšuje. Autorom metódy sečnicových nadrovín je Kelley [4] a preto sa často nazýva aj Kelleyho metóda.

Cieľom práce je podrobne opísať, programovo realizovať a na experimentoch vyhodnotiť metódu sečnicových nadrovín. Výsledky získané z experimentov opíšeme v praktickej časti práce. Za týmto účelom sme najprv programovo realizovali Kelleyho metódu (v jaz. C++) a potom urobili pokusy, na ktorých sme vyhodnotili jej rýchlosť a efektívnosť.

V každej iterácii metódy sečnicových nadrovín sa rieši špeciálna úloha lineárneho programovania a základnou metódou na riešenie úloh lineárneho programovania je simplexová metóda. Preto sa prvá kapitola venuje niek-

torým špecifickým otázkam simplexovej metódy, ktoré súvisia s metódou sečnicových nadrovín a sú potrebné pre naprogramovanie metódy.

V druhej kapitole sa venujeme teoretickému popisu metódy sečnicových nadrovín a budeme riešiť úlohu konvexného programovania s lineárnymi ohraničeniami. Najprv ukážeme základnú myšlienku metódy, v ktorej nelineárnu účelovú funkciu presunieme do ohraničení a potom aproximujeme polyedrickou množinou, čím vzniknú úlohy lineárneho programovania. Na koniec sme metódu sečnicových nadrovín aplikovali a názorne predviedli na konkrétnom príklade.

Kelleyho metódu sme naprogramovali a následne urobili pokusy na sade úloh konvexného programovania s lineárnymi ohraničeniami vytvorenými generátorom úloh. Numerické experimenty vyhodnotia rýchlosť a efektívnosť metódy pre rôzne typy účelových funkcií s meniacim sa počtom premenných a ohraničení.

Základné pojmy a symboly

V tejto časti sa oboznámime s používanými symbolmi pre vektory a matice a potom uvedieme niekoľko definícií. Nakoniec prejdeme k formulácii úlohy konvexného programovania s lineárnymi ohraničeniami, ktorá je predmetom tejto práce.

Stĺpcový vektor označujeme malými písmenami: b, c, x. Riadkový vektor dostaneme transponovaním stĺpcového a značíme: b^T, c^T, x^T. Matice značíme veľkými písmenami: A, C, D.

R je množina reálnych čísel a R_+ je množina nezáporných reálnych čísel. R^n je karteziánska sústava súradníc tvorená z n množín R.

Spolu s vektorom a maticou je spojená informácia o ich rozmere: $x \in R^n$, $A \in R^{m \times n}$. Zápis $x \geq 0_n$ alebo $x \geq 0$ značí, že všetky zložky vektora $x \in R^n$ sú nezáporné.

Spojenie $c^T x$ označuje skalárny súčin dvoch vektorov $c \in R^n$ a $x \in R^n$, teda $c^T x = \sum_{i=1}^{n} c_i x_i$. Súčin matice $A \in R^{m \times n}$ a vektora $x \in R^n$ označujeme Ax.

Def.1 Nech $a \in R^n$, $a \neq 0$, $b \in R$, potom nadrovinou nazývame množinu $N = \{x \in R^n | a^T x = b\}$.

Def.2 Nech $a \in R^n$, $a \neq 0$, $b \in R$, potom polpriestorom nazývame množinu $H = \{x \in R^n | a^T x \leq b\}$.

Def.3 Nech $H_1, \ldots, H_m$ sú polpriestory v R^n, potom množina $M := \bigcap_{i=1}^{m} H_i$ sa nazýva polyedrická množina.

Def.4 Množinu $C \subseteq R^n$ nazývame konvexnou množinou, ak $\forall x, y \in C$ platí, že úsečka daná týmito dvoma bodmi leží celá v C, t.j. $\forall x, y \in C$ a $\lambda \in R$ také, že $0 \leq \lambda \leq 1$ platí

$$\lambda x + (1 - \lambda) y \in C$$

Def.5 Funkcia $f : R^n \to R$ definovaná na konvexnej množine $C \subseteq R^n$ sa nazýva *konvexná funkcia*, ak pre každú dvojicu rôznych bodov $x, y \in C$, $x \neq y$ a ľubovolné číslo $\lambda \in R$, $0 \leq \lambda \leq 1$ platí

$$f(\lambda x + (1 - \lambda) y) \leq \lambda f(x) + (1 - \lambda) f(y)$$

Def.6 Nech $f : R^n \to R$ má parciálne derivácie $\frac{\partial f(x)}{\partial x_i}$, $(i = 1, \ldots, n)$, potom *gradientom* funkcie $f(x)$ je $n-$ rozmerný stĺpcový vektor

$$\nabla f(x) = \left(\frac{\partial f(x)}{\partial x_1}, \frac{\partial f(x)}{\partial x_2}, \cdots, \frac{\partial f(x)}{\partial x_n} \right)^T$$

Def.7 *Úlohou konvexného programovania s lineárnymi ohraničeniami* nazývame úlohu tvaru

$$Min\{f(x) |\ Ax \leq b,\ x \geq 0\} \tag{1}$$

kde $f(x)$ je konvexná funkcia na R_+^n, matica $A \in R^{m \times n}$ a vektor $b \in R^m$.

Množina prípustných riešení úlohy konvexného programovania je množina

$$K = \{x|\ Ax \leq b,\ x \geq 0\}$$

Funkciu $f(x)$ nazývame *účelovou funkciou* a nerovnosti $Ax \leq b$ a $x \geq 0$ nazývame *ohraničeniami*. Riešením úlohy konvexného programovania s lineárnymi ohraničeniami je optimálne riešenie $\hat{x} \in K$, pre ktoré platí $f(\hat{x}) \leq f(x)$, $\forall x \in K$.

Úlohy konvexného programovania sú špecifickými úlohami nelineárneho programovania a majú niektoré osobitné vlastnosti:

1. Každé lokálne minimum konvexnej funkcie je zároveň aj jej globálnym minimom a množina všetkých miním konvexnej funkcie tvorí konvexnú množinu.

2. Množina prípustných riešení úlohy konvexného programovania je konvexná.

Def.8 Pre úlohu (1) definujeme Lagrangeovu funkciu

$$L(x,u) = f(x) + u^T(Ax - b),\ u \geq 0$$

kde vektor $u \in R^m$ sa nazýva vektor Lagrangeových multiplikátorov.

Platí nasledovná *Kuhn-Tuckerova veta*:

Veta. $\hat{x} \geq 0$ je optimálnym riešením úlohy (1) práve vtedy, ak existuje $\hat{u} \geq 0$ tak, že platí:

$$\begin{array}{rclcrcl} \nabla_x L(\hat{x},\hat{u}) & \geq & 0 & & \nabla_u L(\hat{x},\hat{u}) & \leq & 0 \\ \hat{x}^T \nabla_x L(\hat{x},\hat{u}) & = & 0 & & \hat{u}^T \nabla_u L(\hat{x},\hat{u}) & = & 0 \\ \hat{x} & \geq & 0 & & \hat{u} & \geq & 0 \end{array} \qquad (K-T)$$

$(K-T)$ nazývame Kuhn-Tuckerove podmienky.

1 Úloha lineárneho programovania a Simplexova metóda

Metóda sečnicových nadrovín je iteračná metóda riešenia úloh konvexného programovania s lineárnymi ohraničeniami, kde v každej iterácii sa rieši pomocná úloha lineárneho programovania. Pričom pomocné úlohy lineárneho programovania dvoch po sebe nasledujúcich iterácii metódy sečnicových nadrovín sa líšia len pridaním jedného ohraničenia. Najprv sa krátko zoznámime s riešením úloh lineárneho programovania.

Základnou metódou pre riešenie úloh lineárneho programovania je simplexová metóda. Jej autorom je americký matematik G.B. Dantzig a názov simplexová metóda pochádza z prvých úloh, v ktorých množina prípustných riešení bola simplexom. Riešená úloha lineárneho programovania sa zapisuje do tzv. simplexovej tabuľky a ďalej sa upravuje Gauss-Jordanovými eliminániami. Základné pravidlá simplexovej metódy, ktoré sa týkajú riešenia úlohy lineárneho programovania preberieme z [2].

1.1 Základné teoretické poznatky o úlohe lineárneho programovania

V tejto kapitole sa budeme zaoberať simplexovou metódou, ktorá sa používa na riešenie úloh lineárneho programovania. Najprv sa teda oboznámime s lineárnym programovaním a uvedieme úlohu lineárneho programovania.

Úloha lineárneho programovania (LP) je optimalizačná úloha, v ktorej hľadáme maximum alebo minimum lineárnej funkcie pri lineárnych ohraničeniach. Teda v úlohe (LP) hľadáme extrém lineárnej funkcie viazanej podmienkami v tvare lineárnych rovníc a nerovníc. Na oboznámenie sa uvedieme úlohu lineárneho programovania, v ktorej lineárne ohraničenia sú v tvare nerovníc. Skrátene túto úlohu zapíšeme v tvare

$$Min\{c^T x|\ Ax \leq b,\ x \geq 0\} \qquad (LP)$$

kde vektory $x \in R^n$, $c \in R^n$, $b \in R^m$ a $b \geq 0_m$ a matica $A \in R^{m \times n}$. Teda v úlohe (LP) hľadáme minimum lineárnej funkcie

$$c^T x$$

ktorá sa nazýva účelová funkcia, na množine prípustných riešení úlohy lineárneho programovania, čo je množina

$$L = \{x |\ Ax \leq b,\ x \geq 0\}$$

Prípustné riešenie $x^\star \in L$, pre ktoré platí $c^T x^\star \leq c^T x$ pre všetky $x \in L$ nazývame optimálnym riešením.

Vo všeobecnosti môžeme rovnako riešiť maximalizačnú ako aj minimalizačnú úlohu. Keďže pre ľubovoľnú množinu $L \in R^n$ a ľubovoľnú funkciu $f : L \to R$ platí

$$min\{f(x) |\ x \in L\} = -max\{-f(x) |\ x \in L\}$$

môžeme ľubovoľnú minimalizačnú úlohu previesť na maximalizačnú.

Z teórie duality vieme k pôvodnej úlohe lineárneho programovania (LP) priradiť duálnu úlohu (DU). Pôvodná úloha (LP) sa nazýva primárna, potom duálna úloha k úlohe (LP) má tvar

$$Max\{-u^T b |\ A^T u \geq -c,\ u \geq 0\} \qquad (DU)$$

1.2 Optimálne riešenie duálnej úlohy

Predstavme si, že riešime simplexovou metódou primárnu úlohu (LP). Po vyriešení primárnej úlohy získame konečnú optimálnu tabuľku primárnej úlohy, ktorá obsahuje aj optimálne riešenie duálnej úlohy. Ukážeme, že v poslednom riadku tabuľky (riadok účelovej funkcie) v stĺpcoch doplnkových premenných primárnej úlohy sa nachádza optimálne riešenie duálnej úlohy.

Budeme riešiť úlohu lineárneho programovania (LP)

$$Min\{c^T x |\ Ax \leq b,\ x \geq 0\}$$

kde vektory $c \in R^n$, $b \in R^m$ a $b \geq 0_m$ a matica $A \in R^{m \times n}$. Po zavedení pomocnej premennej z a doplnkových premenných y dostaneme úlohu

$$Min \left\{ z \middle| \begin{array}{rcrcl} Iy & + & Ax & = & b, \quad x \geq 0 \\ z & - & c^T x & = & 0, \quad y \geq 0 \end{array} \right\}$$

Túto úlohu dosadíme do štartovaciu simplexovej tabuľky.

pravá strana	y	z	x
b	I	0	A
0	0^T	1	$-c^T$

Ďalej nech $A = (A_B|A_N)$ a $x^T = (x_B^T|x_N^T)$, kde B je množina bázických indexov a N je množina nebázických indexov. Zostavíme podrobnejšiu simplexovú tabuľku

pravá strana	y	z	x_B	x_N
b	I	0	A_B	A_N
0	0^T	1	$-c_B^T$	$-c_N^T$

Nech A_B je optimálna báza, potom budeme robiť nasledovné úpravy pre vyriešenie úlohy simplexovou metódou. Na mieste stĺpcov bázických premenných chceme dostať jednotkovú maticu. Preto prenásobime prvky simplexovej tabuľky maticou A_B^{-1} zľava a urobíme prvú simplexovú iteráciu.

pravá strana	y	z	x_B	x_N
$A_B^{-1}b$	A_B^{-1}	0	I	$A_B^{-1}A_N$
0	0^T	1	$-c_B^T$	$-c_N^T$

Teraz budeme pokračovať v eliminácii premenných x_B, takže k riadku ocenení $-c_B^T$ prirátame c_B^T násobok tabuľky aby sme ho vynulovali

pravá strana	y	z	x_B	x_N
$A_B^{-1}b$	A_B^{-1}	0	I	$A_B^{-1}A_N$
$c_B^TA_B^{-1}b$	$c_B^TA_B^{-1}$	1	0	$c_B^TA_B^{-1}A_N - c_N^T$

Keď už sme eliminovali premenné bázy, budeme predpokladať že všetky ocenenia sú nekladné, čo je kritérium optimality, čiže

$$\begin{aligned} c_B^TA_B^{-1} &\leq 0 \\ c_B^TA_B^{-1}A_N - c_N^T &\leq 0 \end{aligned}$$

keďže $b \geq 0$ tak aj

$$c_B^TA_B^{-1}b \leq 0$$

Tým sme skončili so simplexovými iteráciami a získali sme optimálne riešenie pôvodnej úlohy $\hat{x}$ (kde $\hat{x}_B = A_B^{-1}b$ a $\hat{x}_N = 0$) a optimálnu hodnotu účelovej funkcie $c^T\hat{x} = \hat{z} = c_B^TA_B^{-1}b$, ktoré sa nachádzajú v stĺpci pravej strany. Zároveň sme zo stĺpcov doplnkových premenných y získali optimálne riešenie duálnej úlohy $\hat{u}^T = -c_B^TA_B^{-1}$.

Tvrdenie:
Výraz $\hat{u}^T = -c_B^TA_B^{-1}$
je optimálnym riešením duálnej úlohy

$$Max\{-u^Tb|\ A^Tu \geq -c,\ u \geq 0\} \qquad (DU)$$

Dôkaz:

I) Ukážeme, že $\hat{u}^T = -c_B^T A_B^{-1}$ je prípustné riešenie duálnej úlohy (DU).
Keďže kritérium optimality bolo splnené, a všetky ocenenia sú nekladné, tak $-\hat{u}^T = c_B^T A_B^{-1} \leq 0$, teda $\hat{u}^T \geq 0$.
Teraz ešte ukážeme, že $A^T\hat{u} \geq -c$. Keďže $\hat{u}^T = -c_B^T A_B^{-1}$, tak $\hat{u} = -(A_B^T)^{-1}c_B$. Teraz urobíme úpravu, v ktorej maticu A rozpíšeme na A_B a A_N a za $\hat{u}$ dosadíme $\hat{u} = -(A_B^T)^{-1}c_B$

$$A^T\hat{u} = \begin{pmatrix} A_B^T\hat{u} \\ A_N^T\hat{u} \end{pmatrix} = \begin{pmatrix} -A_B^T(A_B^T)^{-1}c_B \\ -A_N^T(A_B^T)^{-1}c_B \end{pmatrix} = \begin{pmatrix} -c_B \\ -A_N^T(A_B^T)^{-1}c_B \end{pmatrix}$$

Teda $A_B^T\hat{u} = -c_B$ a z kritéria optimality máme $c_B^T A_B^{-1} A_N - c_N^T \leq 0 \Leftrightarrow -\hat{u}^T A_N - c_N^T \leq 0 \Leftrightarrow -\hat{u}^T A_N \leq c_N^T$, teda

$$\begin{aligned} A_B^T\hat{u} &= -c_B \qquad \textit{je splnená} \\ A_N^T\hat{u} &\geq -c_N \qquad \textit{je splnená} \end{aligned}$$

Ukázali sme, že $\hat{u}^T \geq 0$ a $A^T\hat{u} \geq -c$, teda $\hat{u}^T = -c_B^T A_B^{-1}$ je prípustným riešením duálnej úlohy.

II) Ukážeme, že $\hat{u}^T$ je optimálne riešenie duálnej úlohy (DU).
Keďže $c_B^T A_B^{-1} b$ je optimálnou hodnotou účelovej funkcie pôvodnej primárnej úlohy, tak

$$\hat{z} = c^T\hat{x} = c_B^T A_B^{-1} b = -\hat{u}^T b$$

Môžeme použiť aj silnú vetú o dualite, ktorá hovorí, že ak jedna z úloh (primárna alebo duálna) má optimálne riešenie, tak má aj druhá a optimálne hodnoty účelových funkcií sa rovnajú, teda

$$c^T\hat{x} = -\hat{u}^T b$$

Ukázali sme, že v optimálnej simplexovej tabuľke primárnej úlohy v stĺpcoch doplnkových premenných y sa nachádza optimálne riešenie duálnej úlohy, čo budeme používať v metóde sečnicových nadrovín.

1.3 Prvá fáza v prípade rovníc

V tejto časti sa pozrieme na úlohy lineárneho programovania, ktoré majú v ohraničeniach rovnice. Pri riešení týchto úloh najprv zostavíme pomocnú úlohu, ktorá naštartuje simplexovú metódu. Pomocnú úlohu vytvoríme z pôvodnej tak, že pridáme do úlohy umelé premenné a novú účelovú funkciu. Túto úlohu riešime a po skončení dosadíme pôvodnú účelovú funkciu (tomu sa venuje ďalšia časť 1.4).

Teraz ukážeme ako zostaviť k úlohe lineárneho programovania pomocnú úlohu. Nech máme úlohu lineárneho programovania (LP)

$$Min\{c^T x|\ Ax = b,\ x \geq 0_n\}$$

kde vektory $c \in R^n$, $b \in R^m$ a $b \geq 0_m$ a matica $A \in R^{m \times n}$. K nej priradíme pomocnú úlohu, v ktorej $w_1, \ldots, w_m$ sú umelé premenné a na začiatku tvoria umelú bázu.

$$Min\{\sum_{i=1}^{m} w_i|\ Ax + Iw = b,\ x \geq 0_n,\ w \geq 0_m\}$$

Pôvodná úloha má prípustné riešenie práve vtedy, ak pomocná úloha má optimálne riešenie, v ktorom $\sum_{i=1}^{m} w_i = 0$. Ak $\sum_{i=1}^{m} w_i > 0$, potom pôvodná úloha nemá prípustné riešenie.
V prvej fáze teda riešime najprv pomocnú úlohu. Po vyriešení tejto pomocnej úlohy sa štartovacie bázické riešenie

$$\begin{aligned} w &= b \geq 0_m \\ x &= 0_n \end{aligned}$$

v ideálnom prípade nahradí kompletne premennými $x_B > 0_m$. Teda tento ideálny prípad nastáva ak $\sum_{i=1}^{m} w_i = 0$ a v konečnej simplexovej tabuľke sú všetky w_i nebázické. Po skončení prvej fázy sme získali východiskové bázické riešenie pôvodnej úlohy. Stĺpce pomocných premenných w_i môžeme z ďalšieho rátania vynechať. Teraz dosadíme pôvodnú účelovú funkciu a v druhej fáze doriešime úlohu, čím dostaneme výsledok pre pôvodnú úlohu.

1.4 Vyjadrenie riadka účelovej funkcie

Účelovú funkciu zapisujeme do posledného riadka simplexovej tabuľky a vyjadriť posledný riadok simplexovej tabuľky môžeme dvoma spôsobmi. Najprv definujeme úlohu, na ktorej potom oboma spôsobmi odvodíme vyjadrenie posledného riadka. Máme úlohu lineárneho programovania (LP)

$$Min\{c^T x|\ Ax \leq b,\ x \geq 0\}, kde\ b \geq 0$$

Po pridaní doplnkových premenných a definovaním $z = c^T x$ sa úloha prevedie na tvar

$$Min\{0 = z - c^T x|\ Ax + Iy = b,\ x \geq 0,\ y \geq 0\}$$

kde vektory $c \in R^n$, $b \in R^m$ a $b \geq 0_m$ a matica $A \in R^{m \times n}$. Pre túto úlohu zostavíme štartovaciu simplexovú tabuľku

pravá strana	y	z	x_B	x_N
b	I	0	A_B	A_N
0	0^T	1	$-c_B^T$	$-c_N^T$

Pokračujeme riešením štartovacej simplexovej tabuľky, pričom na jej vyriešenie použijeme dve alternatívy. Po doriešení oboch alternatív dostaneme konečnú tabuľku vyjadrenú pomocou premenných zo štartovacej tabuľky a získame aj vyjadrenie posledného riadka účelovej funkcie (vypočítaného cez dve alternatívy).

1. spôsob počítania posledného riadka

Najprv prenásobime prvky východiskovej simplexovej tabuľky okrem posledného riadka maticou A_B^{-1} zľava.

pravá strana	y	z	x_B	x_N
$A_B^{-1}b$	A_B^{-1}	0	I	$A_B^{-1}A_N$
0	0^T	1	$-c_B^T$	$-c_N^T$

Potom k poslednému riadku prirátame c_B^T násobok tabuľky. Po týchto úpravách bude v tabuľke v stĺpci x_B jednotková matica I a posledný riadok tabuľky v stĺpci x_B bude vynulovaný. Dostaneme konečnú optimálnu tabuľku

pravá strana	y	z	x_B	x_N
$A_B^{-1}b$	A_B^{-1}	0	I	$A_B^{-1}A_N$
$c_B^TA_B^{-1}b$	$c_B^TA_B^{-1}$	1	0	$c_B^TA_B^{-1}A_N - c_N^T$

v ktorej platí

$$\begin{aligned} c_B^TA_B^{-1} &\leq 0 \\ c_B^TA_B^{-1}A_N - c_N^T &\leq 0 \end{aligned}$$

keďže $b \geq 0$ tak aj

$$c_B^TA_B^{-1}b \leq 0$$

Prvý spôsob používame pri riešení úloh, ktoré majú v ohraničeniach rovnice a to pri dosadzovaní pôvodnej účelovej funkcie do simplexovej tabuľky po skončení prvej fázy.

2. spôsob počítania posledného riadka

Posledný riadok môžeme počítať cez rozšírenú inverznú bázu, ktorú vypočítame zo stĺpcov x_B a z. Teda zo štartovacej simplexovej tabuľky vyberieme prvky, ktoré sa nachádzajú pod stĺpcami x_B a z. Zostavíme z nich maticu, ktorá má rozmer $(m+1)\times(m+1)$ a vypočítame k nej inverznú maticu

$$\left(\begin{array}{cc|cc} A_B & 0 & 1 & 0 \\ -c_B^T & 1 & 0 & 1 \end{array}\right) \sim \left(\begin{array}{cc|cc} 1 & 0 & A_B^{-1} & 0 \\ -c_B^T & 1 & 0 & 1 \end{array}\right) \sim \left(\begin{array}{cc|cc} 1 & 0 & A_B^{-1} & 0 \\ 0 & 1 & -c_B^T A_B^{-1} & 1 \end{array}\right)$$

Vypočítanou inverznou maticou, ktorú nazveme rozšírená inverzná báza násobime zľava prvky štartovacej simplexovej a aj posledný riadok tabuľky.

$$\left(\begin{array}{c|c} A_B^{-1} & 0 \\ \hline -c_B^T A_B^{-1} & 1 \end{array}\right) *$$

pravá strana	y	z	x_B	x_N
b	I	0	A_B	A_N
0	0^T	1	$-c_B^T$	$-c_N^T$

Dostaneme konečnú optimálnu tabuľku, v ktorej $c_B^T A_B^{-1} \leq 0$ a $c_B^T A_B^{-1} A_N - c_N^T \leq 0$.

pravá strana	y	z	x_B	x_N
$A_B^{-1}b$	A_B^{-1}	0	I	$A_B^{-1}A_N$
$c_B^T A_B^{-1} b$	$c_B^T A_B^{-1}$	1	0	$c_B^T A_B^{-1} A_N - c_N^T$

Všimnime si, že rozšírená inverzná báza sa teraz nachádza v stĺpcoch doplnkových premenných y a premennej z. Ak po získaní optimálnej tabuľky chceme do nej pridať nový stĺpec, pred zaradením tohto stĺpca do tabuľky ho vynásobime rozšírenou inverznou bázou, ktorú už nemusíme počítať a len ju vyberieme z tabuľky.

2 Teória duality v lineárnom programovaní

V lineárnom programovaní môžeme každej minimalizačnej úlohe priradiť maximalizačnú úlohu, ktorá je ňou jednoznačne určená, a podobne k maximalizačnej úlohe vieme priradiť minimalizačnú. Tieto úlohy sa nazývajú primárna úloha a duálna úloha.

Ukážeme ako odvodiť z maximalizačnej primárnej úlohy minimalizačnú duálnu úlohu. Uvažujme úlohu lineárneho programovania tvaru

$$\begin{array}{lrcll} max & \sum_{j=1}^{n} & c_j x_j & & \\ & \sum_{j=1}^{n} & a_{ij} x_j \leq & b_i \quad , \quad i \in I_1 & (A.1.2) \\ & \sum_{j=1}^{n} & a_{ij} x_j = & b_i \quad , \quad i \in I - I_1 & (A.1.3) \\ & & x_j \geq & 0 \quad , \quad j \in J_1 & \end{array} \tag{A.1}$$

kde $x \in R^n$, $c \in R^n$, $b \in R^m$, $a_{ij} \in A_{m \times n}$, $I = \{1, \ldots, m\}$, $J = \{1, \ldots, n\}$ a $I_1 \in I$, $J_1 \in J$.

Pokračujeme tým, že nerovnosť (A.1.2) postupne prenásobíme nezápornými číslami u_i, $i \in I_1$ a rovnice (A.1.3) vynásobime ľubovoľnými číslami u_i, $i \in I - I_1$, ktoré nemusia byť nezáporné. Potom pre každé prípustné riešenie x úlohy (A.1) dostaneme

$$\begin{array}{rcll} \sum_{i=1}^{m} u_i \sum_{j=1}^{n} a_{ij} x_j \leq & \sum_{i=1}^{m} u_i b_i \quad , & i \in I_1 \\ \sum_{i=1}^{m} u_i \sum_{j=1}^{n} a_{ij} x_j = & \sum_{i=1}^{m} u_i b_i \quad , & i \in I - I_1 \\ u_i \geq & 0 \quad , & i \in I_1 \end{array}$$

Zavedieme podmienku, že ak sa podarí zvoliť čísla u_i, $i \in I$ tak, aby platilo

$$\begin{array}{rcl} u_1 a_{11} + u_2 a_{21} + \ldots + u_m a_{m1} & \geq & c_1 \\ \vdots & & \vdots \\ u_1 a_{1k} + u_2 a_{2k} + \ldots + u_m a_{mk} & \geq & c_k \\ u_1 a_{1k+1} + u_2 a_{2k+1} + \ldots + u_m a_{mk+1} & = & c_{k+1} \\ \vdots & & \vdots \\ u_1 a_{1n} + u_2 a_{2n} + \ldots + u_m a_{mn} & = & c_n \end{array}$$

kde $k \in J$. Teda skrátene môžeme napísať

$$\begin{array}{rcll} \sum_{i=1}^{m} u_i a_{ij} & \geq & c_j & , \quad j \in J_1 \\ \sum_{i=1}^{m} u_i a_{ij} & = & c_j & , \quad j \in J - J_1 \end{array}$$

Potom vďaka nezápornosti prípustného riešenia x úlohy (A.1), po prenásobení sústavy týmto riešením, dostávame

$$\begin{array}{rcll} \sum_{j=1}^{n} x_j \sum_{i=1}^{m} u_i a_{ij} & \geq & \sum_{j=1}^{n} c_j x_j, & j \in J_1 \\ \sum_{j=1}^{n} x_j \sum_{i=1}^{m} u_i a_{ij} & = & \sum_{j=1}^{n} c_j x_j, & j \in J - J_1 \end{array}$$

Keďže platí rovnosť

$$\sum_{j=1}^{n} x_j \sum_{i=1}^{m} u_i a_{ij} = \sum_{i=1}^{m} u_i \sum_{j=1}^{n} a_{ij} x_j$$

dostávame

$$\begin{array}{l} \sum_{j=1}^{n} c_j x_j \leq \sum_{j=1}^{n} x_j \sum_{i=1}^{m} u_i a_{ij} = \sum_{i=1}^{m} u_i \sum_{j=1}^{n} a_{ij} x_j \leq \sum_{i=1}^{m} u_i b_i, \quad j \in J_1 \\ \sum_{j=1}^{n} c_j x_j = \sum_{j=1}^{n} x_j \sum_{i=1}^{m} u_i a_{ij} = \sum_{i=1}^{m} u_i \sum_{j=1}^{n} a_{ij} x_j = \sum_{i=1}^{m} u_i b_i, \quad j \in J - J_1 \end{array}$$

Vidíme, že číslo $\sum_{i=1}^{m} u_i b_i$ je pre našu riešenú úlohu horným odhadom hodnôt účelovej funkcie na množine prípustných riešení, keďže

$$\sum_{j=1}^{n} c_j x_j \leq \sum_{i=1}^{m} u_i b_i$$

Takže, keď sme v pôvodnej úlohe riešili maximalizáciu a hľadali sme maximálnu hodnotu účelovej funkcie $\sum_{j=1}^{n} c_j x_j$, teraz budeme riešiť minimalizačnú úlohu, v ktorej budeme hľadať minimálnu hodnotu účelovej funkcie $\sum_{i=1}^{m} u_i b_i$. Tým sa dostávame k úlohe

$$\begin{array}{lrcll} min & \sum_{i=1}^{m} u_i b_i & & & \\ & \sum_{i=1}^{m} u_i a_{ij} & \geq & c_j \quad , & j \in J_1 \\ & \sum_{i=1}^{m} u_i a_{ij} & = & c_j \quad , & j \in J - J_1 \\ & u_i & \geq & 0 \quad , & i \in I_1 \end{array} \qquad (A.2)$$

Táto minimalizačná úloha sa nazýva *duálna úloha* k pôvodnej maximalizačnej úlohe (A.1), ktorá sa nazýva *primárna úloha*. Pretože názvy primárna a duálna závisia od toho, z ktorej úlohy vychádzame na začiatku, môžeme hovoriť o *dvojici vzájomne duálnych úloh*.

Príklad. Špeciálnym prípadom úlohy (A.1), ktorá vznikne ak $I_1 = I$ a $J_1 \neq \emptyset, J_1 \in J$ je úloha tvaru

$$\begin{array}{lrcll} max & \sum_{j=1}^{n} & c_j x_j & & \\ & \sum_{j=1}^{n} & a_{ij} x_j & \leq \; b_i \quad , & i \in I \\ & & x_j & \geq \; 0 \quad , & j \in J_1 \end{array} \tag{A.3}$$

V tejto úlohe sa na rozdiel od predchádzajúcej nachádzajú iba rovnosti. J_1 označuje počet nezáporných premenných a $J - J_1$ počet voľných premenných. Duálna úloha k nej má tvar

$$\begin{array}{lrcll} min & \sum_{i=1}^{m} & u_i b_i & & \\ & \sum_{i=1}^{m} & u_i a_{ij} & \geq \; c_j \quad , & j \in J_1 \\ & \sum_{i=1}^{m} & u_i a_{ij} & = \; c_j \quad , & j \in J - J_1 \\ & & u_i & \geq \; 0 \quad , & i \in I \end{array} \tag{A.4}$$

Nasledujúce dve vety hovoria o vzťahoch medzi riešením primárnej a riešením duálnej úlohy.

Veta 1. Slabá veta o dualite. Pre každé prípustné riešenie $\bar{x}$ primárnej úlohy (A.1) a každé prípustné riešenie $\bar{u}$ duálnej úlohy (A.2) platí

$$c^T \bar{x} \leq b^T \bar{u}$$

Veta 2. Silná veta o dualite. Ak jedna z dvojice vzájomne duálnych úloh má optimálne riešenie, tak má optimálne riešenie aj druhá a optimálne hodnoty účelových funkcií sa rovnajú.

3 Metóda sečnicových nadrovín

Metóda sečnicových nadrovín je iteratívnou metódou riešenia úloh konvexného programovania s lineárnymi ohraničeniami. Riešenie úloh pozostáva z iterácii, kde v každej iterácii sa rieši úloha lineárneho programovania, pričom po každej iterácii sa príslušná úloha lineárneho programovania rozširuje o ďalšie ohraničenie. Pridané ohraničenie sa nazýva sečnicová nadrovina (z angl. Cutting Plane), z čoho je vlastne odvodené meno metódy. Keďže dve po sebe nasledujúce úlohy lineárneho programovania sa navzájom líšia len pridaním jedného ohraničenia, je výhodné riešiť ich ako duálne úlohy, ktoré sa budú od seba líšiť len jedným stĺpcom. Potom optimálne riešenie z prvej úlohy sa použije ako východiskové riešenie pre nasledujúcu úlohu. Autorom metódy sečnicových nadrovín je Kelley[4] a preto sa často nazýva aj ako Kelleyho metóda.

3.1 Základná myšlienka metódy sečnicových nadrovín

Metóda sečnicových nadrovín rieši úlohu konvexného programovania s lineárnymi ohraničeniami. Takúto úlohu vieme pretransformovať na inú úlohu, v ktorej účelová funkcia bude lineárna a z pôvodnej nelineárnej účelovej funkcie urobíme nové ohraničenie. Potom konvexnú množinu určenú týmto nelineárnym ohraničením postupne aproximujeme polyedrickou množinou, čím vzniknú úlohy lineárneho programovania.

Budeme riešiť úlohu konvexného programovania tvaru

$$Min\{f(x)|\ Ax \leq b,\ x \geq 0\} \tag{1}$$

s množinou prípustných riešení

$$K = \{x \in R^n|\ Ax \leq b,\ x \geq 0\}$$

kde $f(x)$ je konvexná funkcia so spojitými prvými parciálnymi deriváciami na K a matica $A \in R^{m\times n}$, vektor $b \in R^m$.

Úlohu (1) chceme nejakým spôsobom linearizovať, t.j. jedinú nelineárnu funkciu potrebujeme nahradiť po častiach lineárnou funkciou. To urobíme presunutím účelovej funkcie $f(x)$ do ohraničení a potom príslušnú množinu ohraničení nahradíme aproximujúcou polyedrickou množinou.

Úlohu (1) môžeme pretransformovať pridaním ďalšej premennej ξ na ekvivalentnú úlohu

$$Min\{\xi|\ f(x) \leq \xi,\ x \in K\} \tag{2}$$

Teda ak $\hat{x}$ je optimálne riešenie úlohy (1) a $\hat{\xi} = f(\hat{x})$, potom $(\hat{\xi}, \hat{x})$ je optimálnym riešením úlohy (2) a naopak, ak $(\hat{\xi}, \hat{x})$ je optimálnym riešením úlohy (2), potom $\hat{\xi} = f(\hat{x})$ a $\hat{x}$ je optimálnym riešením úlohy (1).
Množina prípustných riešení úlohy (2) je množina

$$M = \{(x, \xi) \in R^{n+1}|\ f(x) \leq \xi,\ x \in K\}$$

Keďže $f(x)$ je konvexná funkcia na množine K, tak množina M, teda množina prípustných riešení úlohy (2) je konvexnou množinou v R^{n+1}. Túto množinu v najjednoduchšom prípade môžeme aproximovať nasledovnou polyedrickou množinou

$$M = \{(x, \xi) \in R^{n+1}|\ x \in K,\ g_*^T x - \gamma_* \leq \xi\}$$

kde $x^* \in K$, $g_* = \nabla f(x^*)$, $\gamma_* = \nabla f(x^*)x^* - f(x^*)$ a $\nabla f(x)$ je gradient funkcie $f(x)$ v x^*. Rovnica $g_*^T x - \gamma_* = y$ je rovnica dotykovej nadroviny ku grafu funkcie $f(x)$ prechádzajúca bodom $[x^*, f(x^*)] \in R^{n+1}$.

Opakovaním uvedenej aproximácie môžeme vytvoriť r dotykových nadrovín k funkcii $f(x)$ a množinu prípustných riešení úlohy (2) M môžeme aproximovať množinou vytvorenou z r jej dotykových nadrovín. Teda nech $x^k \in K$, $k = 0, 1, \ldots, r-1$. Definujeme g_k a γ_k, $k = 0, 1, \ldots, r-1$ vzťahmi

$$\begin{aligned} g_k &= \nabla f(x^k) \\ \gamma_k &= \nabla f(x^k)x^k - f(x^k) \end{aligned} \tag{3}$$

Rovnica dotykovej nadroviny ku grafu funkcie $f(x)$ prechádzajúca jej bodom $[x^k, f(x^k)]$ má tvar

$$y = g_k^T x - \gamma_k$$

Teda ak je daných r bodov $x^k \in K,\ k = 0, 1, \ldots, r-1$, tak množinu prípustných riešení úlohy (2) M môžeme aproximovať množinou

$$M(r) = \{(x, \xi) \in R^{n+1} |\ x \in K,\ g_k^T x - \gamma_k \leq \xi,\ k = 0, 1, \ldots, r-1\}$$

Aproximácia bude tým presnejšia, čím hustejšia bude sieť bodov $x^k \in K$. Ak pridáme k bodom $x^k,\ k = 0, 1, \ldots, r-1$ ďalší bod $x^r \in K$ potom dotyková nadrovina $y = g_r x - \gamma_r$ odsekne časť množiny $M(r)$, teda $M(r+1) \subset M(r)$.

Z tohoto postupu vyplýva, že ak chceme získať približné riešenie úlohy (2) a tým aj približné riešenie úlohy (1), možno ho získať riešením tejto úlohy lineárneho programovania:

$$Min\{\xi |\ x \in K,\ g_k^T x - \gamma_k \leq \xi,\ k = 0, 1, \ldots, r-1\} \tag{4}$$

kde $x^k \in K,\ k = 0, 1, \ldots, r-1$ a $g_k,\ \gamma_k$ sú určené vzťahmi (3). Pridávaním ďalších bodov $x^r, x^{r+1}, \ldots \in K$ možno približné riešenie zlepšovať.

Z hľadiska nášho cieľa vyriešiť úlohu (1), resp. úlohu (2) rozhodujúcou je aproximácia množiny M v okolí optimálneho riešenia $[\hat{x}, f(\hat{x})]$ úlohy (2). Teda zdá sa byť rozumné za ďalší bod x^r zvoliť optimálne riešenie úlohy (4). Tým sme definovali pravidlo pre generovanie postupnosti riešení $x^k \in K$, pričom za začiatočný bod zvolíme $x^0 \in K$.

Keďže (4) je úlohou lineárneho programovania, jej optimálne riešenie (x, ξ) je krajným bodom polyedrickej množiny $M(r)$. Množinu $M(r)$ tvoria dotykové nadroviny $g_k^T x - \gamma_k$ a teda bod (x^r, ξ^r),určený optimálnym riešením úlohy (4), leží na jednej z dotykových nadrovín. Z toho že $f(x)$ je konvexná vyplýva, že graf funkcie $f(x)$ leží nad svojimi dotykovými nadrovinami, teda pre všetky $x \in K$ platí

$$f(x) \geq g_k^T x - \gamma_k$$

To znamená, že $f(x) \geq \xi^r$ pre všetky $x \in K$.

Predpokladajme, že postupnosti x^k a $\xi^k,\ k = 0, 1, \ldots, r-1$ sú generované riešením úlohy (4). Vzhľadom na to, že úloha (4) riešená v $(r-1)-$ iterácii sa líši od úlohy (4) riešenej v $r-$ iterácii len pridaním ďalšieho ohraničenia, tak pre postupnosti x^k a $\xi^k,\ k = 0, 1, \ldots, r-1$ platí

$$\xi^1 \leq \xi^2 \leq \ldots \leq \xi^r \leq f(\hat{x}) \leq min\ f(x^k),\ k = 0, 1, \ldots, r \tag{5}$$

Nerovnosti (5) môžeme použiť na odhad presnosti priebežného riešenia. Poznamenáme, že odhad optimálnej hodnoty $f(\hat{x})$ je zdola monotónny a zhora nie. Odhad presnosti priebežného riešenia $r-$ iterácie x^r označíme Δ_r

$$\Delta_r = min\ f(x^r) - \xi^r \geq 0$$

Teda x^r bude Δ presným optimálnym riešením úlohy (1), ak $0 \leq \Delta_r \leq \Delta$, kde $\Delta > 0$ je vopred určený stupeň požadovaný presnosti.

3.2 Realizácia r- tej iterácie Metódy sečnicových nadrovín

V tejto časti preberieme riešenie úlohy (1) metódou sečnicových nadrovín od prvej po $r-$ iteráciu. V každej iterácii sa rieši špeciálna úloha lineárneho programovania. Úlohy lineárneho programovania riešené v $(r-1)-$ iterácii a v $r-$ iterácii sa navzájom líšia len pridaním jedného ohraničenia.

Budeme riešiť úlohu konvexného programovania s lineárnymi ohraničeniami

$$Min\{f(x)|\ Ax \leq b,\ x \geq 0_n\} \tag{6}$$

kde A je matica typu $m \times n$, $x \in R^n$ a $b \in R^m$, $f(x)$ je konvexná funkcia so spojitými prvými parciálnymi deriváciami. Úlohu (1) pretransformujeme pridaním pomocnej premennej z na ekvivalentnú úlohu

$$Min \left\{ z \middle| \begin{array}{lll} Ax & \leq & b, \quad x \geq 0_n \\ f(x) & \leq & z \end{array} \right\} \tag{7}$$

V novej úlohe máme pridané nelineárne ohraničenie v tvare

$$-z + f(x) \leq 0 \tag{8}$$

Toto nelineárne ohraničenie nahradíme systémom lineárnych ohraničení, ktoré konštruujeme ako dotykové nadroviny ku grafu funkcie $f(x)$. Inak povedané funckiu $f(x)$ aproximujeme prvými dvoma členmi jej Taylorovho rozvoja v bode x^k, t.j.

$$f(x) \approx f(x^k) + g_k^T(x - x^k)$$

kde rovnica $y = f(x^k) + g_k^T(x - x^k)$ reprezentuje dotykovú nadrovinu v bode x^k ku grafu funkcie $f(x)$.
Teda nelineárne ohraničenie (8) nahradíme systémom lineárnych ohraničení

$$-z + f(x^k) + g_k^T(x - x^k) \leq 0,\ k = 0, 1, \ldots, r-1$$

ktoré môžeme použitím vzťahov (3) upraviť na tvar

$$-z + g_k^T x \leq \gamma_k$$

Týmto spôsobom sme nelineárnu úlohu (7) aproximovali nasledovnou úlohou lineárneho programovania

$$Min \left\{ 1 \cdot z + 0^T x \,\middle|\, \begin{array}{rcccl} & & Ax & \leq & b, \quad x \geq 0_n \\ -z & + & g_k^T x & \leq & \gamma_k, \quad k = 0, 1, \ldots, r-1 \end{array} \right\} \tag{9}$$

Úlohu (9) po zavedení substitúcie $y = -z$ a zmene úlohy na maximalizáciu pretransformujeme na úlohu

$$-Max \left\{ 1 \cdot y - 0^T x \,\middle|\, \begin{array}{rcccl} & & Ax & \leq & b \\ & y & + & g_k^T x & \leq \gamma_k \\ y \in R, & x \geq 0_n, & & k = 0, 1, \ldots, r-1 & \end{array} \right\} \tag{10}$$

Vzhľadom na to, že úlohy lineárneho programovania (10) prislúchajúce po sebe nasledujúcim iteráciám $(r-1)$ a r sa navzájom líšia len pridaním jedného ohraničenia, je výhodné riešiť simplexovým algoritmom ich duálne úlohy. V danom prípade sa dve duálne úlohy navzájom líšia len pridaním jedného stĺpca, a tak optimálne riešenie predchádzajúcej úlohy môže slúžiť ako východiskové bázické riešenie nasledujúcej úlohy.

Konštrukciou duálnej úlohy sme sa zaoberali v kapitole 2. S využitím toho, že úloha (10) je špeciálnym tvarom úlohy (A.3) kapitoly 2, duálna úloha k úlohe (10) má podľa (A.4) tvar

$$-Min \left\{ b^T u + \sum_{k=0}^{r-1} \gamma_k v_k \,\middle|\, \begin{array}{rcccl} A^T u & + & \sum_{k=0}^{r-1} v_k g_k & \geq & 0_n, \quad u \geq 0 \\ & & \sum_{k=0}^{r-1} v_k & = & 1, \quad v \geq 0 \end{array} \right\} \tag{11}$$

resp.

$$-Min\left\{ b^T u + \sum_{k=0}^{r-1} \gamma_k v_k \,\middle|\, \begin{array}{rcccccc} -A^T u & - & \sum_{k=0}^{r-1} & v_k g_k & \leq & 0_n, & u \geq 0 \\ & & \sum_{k=0}^{r-1} & v_k & = & 1, & v \geq 0 \end{array} \right\} \quad (12)$$

Po zavedení n doplnkových premenných $w \geq 0_n$ dostávame úlohu, vhodnú na zostavenie prvej simplexovej tabuľky. Označíme ju ako úloha (12a).

$$-Min\left\{ b^T u + \sum_{k=0}^{r-1} \gamma_k v_k + 0^T w \,\middle|\, \begin{array}{rcccccc} -A^T u & - & \sum_{k=0}^{r-1} & v_k g_k & + & Iw & = & 0_n \\ & & \sum_{k=0}^{r-1} & v_k & & & = & 1 \\ & & u \geq 0_m, & v \geq 0_r, & & w \geq 0_n & & \end{array} \right\}$$

Simplexová tabuľka, do ktorej zapíšeme jednotlivé koeficienty úlohy (12a) bude mať pre $r = 1$ nasledovný tvar (pozri nižšie). V ľavom krajnom stĺpci sú koeficienty pravých strán, potom nasledujú stĺpce doplnkových premenných w, premennej z, premenných u a v pravom krajnom stĺpci je záporný gradient $-g_0 = -\nabla f(x^0)$.

V hornom riadku sú označenia premenných, ktorým náleží daný stĺpec, v predposlednom riadku sú koeficienty rovnice $\sum_{k=0}^{r-1} v_k = 1$ a v dolnom riadku sú zapísané koeficienty účelovej funkcie, teda ocenenia.

pravá strana	w	z	u	v_0
0	I	0	$-A^T$	$-g_0$
1	0^T	0	0^T	1
0	0^T	1	b^T	γ_0

K štartu simplexovej metódy potrebujeme zaviesť jednu pomocnú premennú ω, ktorú v prvej fáze simplexovej metódy budeme eliminovať. Pre tento účel vytvoríme pomocnú úlohu

$$-Min\left\{ 1 \cdot \omega \,\middle|\, \begin{array}{rcccccccc} -A^T u & - & \sum_{k=0}^{r-1} & v_k g_k & + & Iw & = & 0_n, & u \geq 0_m, \quad w \geq 0_n \\ & & \sum_{k=0}^{r-1} & v_k & + & \omega & = & 1, & v \geq 0_r, \quad \omega \geq 0 \end{array} \right\}$$

Z tejto pomocnej úlohy zostavíme východiskovú simplexovú tabuľku prvej fázy, ktorá je zhodná s predchádzajúcou, a naviac obsahuje stĺpec pomocnej premennej ω a zmenený posledný riadok tabuľky, ktorý predstavuje účelovú funkciu.

pravá strana	w	ω	z	u	v_0
0	I	0	0	$-A^T$	$-g_0$
1	0^T	1	0	0^T	1
0	0^T	1	1	0^T	0

Riešime prvú fázu simplexovej metódy a po skončení dosadíme pôvodnú účelovú funkciu $z = b^T u + \gamma_0 v_0 + 0^T w$ do posledného riadka tabuľky. Upravíme posledný riadok prvým spôsobom z časti (1.4) a naštartujeme druhú fázu simplexovej metódy.

Po skončení druhej fázy získame optimálne riešenie x^1. Tým sme skončili proces prvej iterácie metódy sečnicových nadrovín a budeme pokračovať druhou iteráciou. Vyrátame $g_1 = \nabla f(x^1), \gamma_1 = \nabla f(x^1)x^1 - f(x^1)$ a vytvoríme nový stĺpec v_1.

$$v_1 = \begin{pmatrix} -g_1 \\ 1 \\ -\gamma_1 \end{pmatrix}$$

Stĺpec v_1 pridáme do simplexovej tabuľky, ktorú sme získali po skončení druhej fázy simplexovej metódy. Pred zavedením tohto stĺpca do tabuľky ho vynásobime rozšírenou inverznou bázou, ktorá sa nachádza v stĺpcoch w, ω, z.

$$\begin{matrix} w & \omega & z \end{matrix} \qquad v_1 \qquad\qquad v_1$$

$$\begin{pmatrix} X & U & 0 \\ & & \\ Y & Z & 1 \end{pmatrix} * \begin{pmatrix} -g_1 \\ 1 \\ -\gamma_1 \end{pmatrix} = \begin{pmatrix} -\bar{g}_1 = X * (-g_1) + U * 1 - 0 * \gamma_1 \\ \\ -\bar{\gamma}_1 = Y * (-g_1) + Z * 1 - 1 * \gamma_1 \end{pmatrix}$$

Takto upravený stĺpec v_1 pridáme do simplexovej tabuľky z konca druhej fázy. Pokračujeme v riešení rozšírenej simplexovej tabuľky s pridaným stĺpcom v_1.

Po doriešení rozšírenej simplexovej tabuľky získame optimálne riešenie x^2 a skončili sme druhú iteráciu metódy sečnicových nadrovín. Ďalej pokračujeme treťou iteráciou. Vyrátame nový stĺpec v_2, ktorý vynásobime rozšírenou inverznou bázou zo stĺpcov w, ω, z (získanými z tabuľky z konca druhej iterácie). Riešime simplexovú tabuľku rozšírenú o stĺpec v_2.

Na konci $r-$ tej iterácie metódy sečnicových nadrovín získame optimálne riešenie x^r. V $(r+1)-$ iterácii určíme $g_r = \nabla f(x^r)$, $\gamma_r = \nabla f(x^r)x^r - f(x^r)$ a do tabuľky z konca $r-$ tej iterácie pridáme stĺpec novej premennej v_r

$$v_r = \begin{pmatrix} -g_r \\ 1 \\ -\gamma_r \end{pmatrix} = \begin{pmatrix} -\nabla f(x^r) \\ 1 \\ -\nabla f(x^r)x^r + f(x^r) \end{pmatrix}$$

Predtým ako ho pridáme do tabuľky a budeme riešiť $(r+1)-$ iteráciu, musíme ho vynásobiť rozšírenou inverznou bázou, ktorá sa nachádza v konečnej simplexovej tabuľke $r-$ tej iterácie na mieste stĺpcov premenných w, ω a z. Takto upravený stĺpec v_r dosadíme do simplexovej tabuľky získanej na konci $r-$ tej iterácie a dostaneme východiskovú simplexovú tabuľku na riešenie $(r+1)-$ iterácie.

pravá strana	w	ω	z	u	v_0	v_1	$\cdots$	v_r

3.3 Zhrnutie schémy algoritmu

Postup riešenia úlohy metódou sečnicových nadrovín zhrnieme do nasledovného algoritmu.
Budeme riešiť úlohu konvexného programovania s lineárnymi ohraničeniami

$$Min\{f(x)|\ Ax \leq b,\ x \geq 0\} \tag{1}$$

Na začiatku zvolíme bod $x^0 \geq 0$ tak, aby $Ax^0 \leq b$.
Stanovíme presnosť $\Delta > 0$ a po skončení každej iterácie vyrátame presnosť priebežného riešenia

$$\Delta_k = min\{f(x^k)\} - z^k$$

Tým budeme kontrolovať presnosť priebežného riešenia. Teda v $r-$tej iterácii vyrátame Δ_r a ak $\Delta_r > \Delta$ pokračujeme v $(r+1)-$ iterácii, ak $\Delta_r < \Delta$, dosiahli sme požadovanú presnosť a skončíme počítanie.

1.iterácia:
Vyrátame $g_0 = \nabla f(x^0)$, $\gamma_0 = \nabla f(x^0)x^0 - f(x^0)$ a riešime úlohu (12) pre $r = 1$, teda úlohu

$$Min\left\{ b^T u + \gamma_0 v_0 \middle| \begin{array}{rcrcll} -A^T u & - & v_0 g_0 & \leq & 0_n, & u \geq 0 \\ & & v_0 & = & 1, & v_0 \geq 0 \end{array} \right\}$$

Zavedieme n doplnkových premenných a jednu pomocnú premennú ω a zostavíme pomocnú úlohu.

$$Min\left\{ \omega \middle| \begin{array}{rcrcrcll} -A^T u & - & v_0 g_0 & + & w & = & 0_n, & u \geq 0_m, \quad w \geq 0_n \\ & & v_0 & + & \omega & = & 1, & v_0 \geq 0_r, \quad \omega \geq 0 \end{array} \right\}$$

Pomocnú úlohu dosadíme do simplexovej tabuľky a riešime prvú fázu simplexovej metódy.

pravá strana	w	ω	z	u	v_0
0	I	0	0	$-A^T$	$-g_0$
1	0^T	1	0	0^T	1
0	0^T	1	1	0^T	0

Po skončení prvej fázy simplexovej metódy získame získame východiskové bázické riešenie úlohy (12) pre $r = 1$. Dosadíme pôvodnú účelovú funkciu a v druhej fáze doriešime. Po vyriešení tejto úlohy získame v stĺpcoch doplnkových premenných w optimálne riešenie x^1 úlohy (10) resp. (4) a $z^1 = -y^1 = -min\{\ b^T u + \gamma_0 v_0\ \}$. Skontrolujeme presnosť a ak sme nedosiahli požadovanú presnosť pokračujeme v druhej iterácii.

2.iterácia:
Určíme $g_1 = \nabla f(x^1)$, $\gamma_1 = \nabla f(x^1)x^1 - f(x^1)$. Určíme stĺpec premennej $v_1 = (-g_1, 1, -\gamma_1)^T$ a vynásobime ho rozšírenou inverznou bázou, ktorá sa nachádza v stĺpcoch premenných w, ω, z. Potom ho pridáme do tabuľky získanej na konci prvej iterácie a riešime druhú iteráciu.
Po pridaní nového stĺpca v_1 (podľa str. 29) sme dostali rozšírenú simplexovú tabuľku

pravá strana	w	ω	z	u	v_0	v_1
0	X	U	0			$-\bar{g}_1$
	Y	Z	1			$-\bar{\gamma}_1$

Po vyriešení simplexovej tabuľky získame optimálne riešenie x^2 a pokiaľ sme nedosiahli požadovanú presnosť, pokračujeme v riešení opakovaním postupu druhej iterácie.

3.4 Ilustratívny príklad riešenia úlohy Metódou sečnicových nadrovín

Zadanie.

Riešme metódou sečnicových nadrovín úlohu konvexného programovania s lineárnymi ohraničeniami

$$Min\{f(x)|\ Ax \leq b,\ x \geq 0\}$$

S počtom premenných $n = 2$ a počtom ohraničení $m = 2$. Účelová funkcia $f(x)$ je logaritmická, tvaru $f(x) = \sum_{i=1}^{n} x_i ln(x_i) + p^T x,\ (i = 1, \ldots, n)$. Pričom ak $x_i = 0$, potom $x_i ln(x_i) = 0$. Vektory $p \in R^n$, $b \in R^m$ a $b \geq 0_m$ a matica $A \in R^{m \times n}$.

$$A = \begin{pmatrix} 2 & 7 \\ 1 & 5 \end{pmatrix}, \quad b = \begin{pmatrix} 31 \\ 20 \end{pmatrix}, \quad p = \begin{pmatrix} -ln(5) - 10 \\ -ln(3) - 34 \end{pmatrix}$$

Optimálne riešenie tejto úlohy je $\hat{x} = (5, 3)^T$ a $f(\hat{x}) = -152$. Za východiskový bod zvolíme $x^0 = (0, 0)^T$.

Postup riešenia.

Najprv určíme $g_k = \nabla f(x^k) = \begin{pmatrix} \frac{\partial f(x^1)}{\partial x_1^1} \\ \frac{\partial f(x^1)}{\partial x_2^1} \end{pmatrix} = \begin{pmatrix} \frac{\partial (\sum_{i=1}^{n} x_i^k ln(x_i^k) + p^T x^k)}{\partial x_1^k} \\ \frac{\partial (\sum_{i=1}^{n} x_i^k ln(x_i^k) + p^T x^k)}{\partial x_2^k} \end{pmatrix}$ a $\gamma_k = \nabla f(x^k) x^k - f(x^k)$

1.iterácia.

Zo štartovacieho bodu $x^0 = (0, 0)^T$ vyrátame hodnoty premenných g_0 a γ_0.

$$g_0 = \nabla f(x^0) = \begin{pmatrix} -ln(5) - 10 \\ -ln(3) - 34 \end{pmatrix}, \quad \gamma_0 = \nabla f(x^0) x^0 - f(x^0) = 0$$

Zostavíme úlohu (12) pre $r = 1$, teda úlohu

$$Min\left\{ 31u_1 + 20u_2 + 0v_0 \;\middle|\; \begin{array}{rrcl} -2u_1 - 1u_2 & +(ln(5)+10)v_0 & \leq & 0_n \\ -7u_1 - 5u_2 & +(ln(3)+34)v_0 & \leq & 0_n \\ & v_0 & = & 1 \\ u \geq 0, & v_0 \geq 0 & & \end{array} \right\}$$

K tejto úlohe vytvoríme pomocnú úlohu, ktorá naštartuje proces riešenia pôvodného príkladu. Je to pomocná úloha riešená v prvej fáze simplexovej metódy.

$$Min\left\{ \omega \;\middle|\; \begin{array}{rrrcl} -2u_1 - 1u_2 & +(ln(5)+10)v_0 & & \leq & 0_n \\ -7u_1 - 5u_2 & +(ln(3)+34)v_0 & & \leq & 0_n \\ & v_0 & +\omega & = & 1 \\ u \geq 0, & v_0 \geq 0, & \omega \geq 0 & & \end{array} \right\}$$

Zostavíme simplexovú tabuľku, do ktorej dosadíme hodnoty pomocnej úlohy a riešime prvú fázu simplexovej metódy.

pravá strana	w_1	w_2	ω	u_1	u_2	v_0
0	1	0	0	-2	-1	$ln(5)+10$
0	0	1	0	-7	-5	$ln(3)+34$
1	0	0	1	0	0	1
0	0	0	1	0	0	0

Po prvom kroku prvej fázy dostaneme tabuľku

pravá strana	w_1	w_2	ω	u_1	u_2	v_0
0	1	0	0	-2	-1	$ln(5)+10$
0	0	1	0	-7	-5	$ln(3)+34$
1	0	0	1	0	0	1
1	0	0	0	0	0	1

Za pivota volíme prvok v stĺpci v_0 v druhom riadku.

pravá strana	w_1	w_2	ω	u_1	u_2	v_0
0	1	$-\frac{ln(5)+10}{ln(3)+34}$	0	$\frac{-2ln(3)+7ln(5)+2}{ln(3)+34}$	$\frac{-ln3+5ln(5)+16}{ln(3)+34}$	0
0	0	$\frac{1}{ln(3)+34}$	0	$-\frac{7}{ln(3)+34}$	$-\frac{5}{ln(3)+34}$	1
1	0	$-\frac{1}{ln(3)+34}$	1	$\frac{7}{ln(3)+34}$	$\frac{5}{ln(3)+34}$	0
1	0	$-\frac{1}{ln(3)+34}$	0	$\frac{7}{ln(3)+34}$	$\frac{5}{ln(3)+34}$	0

Za pivota volíme prvok v stĺpci u_1 v prvom riadku.

pravá strana	w_1	w_2	ω	u_1	u_2	v_0
0	$\frac{ln(3)+34}{-2ln(3)+7ln(5)+2}$	$\frac{-ln(5)-10}{-2ln(3)+7ln(5)+2}$	0	1	$\frac{-ln3+5ln(5)+16}{-2ln(3)+7ln(5)+2}$	0
0	$\frac{7}{-2ln(3)+7ln(5)+2}$	$\frac{-2}{-2ln(3)+7ln(5)+2}$	0	0	$\frac{3}{-2ln(3)+7ln(5)+2}$	1
1	$\frac{-7}{-2ln(3)+7ln(5)+2}$	$\frac{2}{-2ln(3)+7ln(5)+2}$	1	0	$\frac{-3}{-2ln(3)+7ln(5)+2}$	0
1	$\frac{-7}{-2ln(3)+7ln(5)+2}$	$\frac{2}{-2ln(3)+7ln(5)+2}$	0	0	$\frac{-3}{-2ln(3)+7ln(5)+2}$	0

Za pivota volíme prvok v stĺpci w_2 v treťom riadku.

pravá strana	w_1	w_2	ω	u_1	u_2	v_0
$\frac{ln(5)+10}{2}$	$-\frac{1}{2}$	0	$\frac{ln(5)+10}{2}$	1	$\frac{1}{2}$	0
1	0	0	1	0	0	1
$\frac{-2ln(3)+7ln(5)+2}{2}$	$-\frac{7}{2}$	1	$\frac{-2ln(3)+7ln(5)+2}{2}$	0	$-\frac{3}{2}$	0
0	0	0	-1	0	0	0

Po skončení prvej fázy simplexovej metódy dosadíme do posledného riadka pôvodnú účelovú funkciu. Stĺpec omega vynechávame z ďalšieho rátania. Z tabuľky vidíme, že vektor bázických premenných, označíme ho $x_B = (u_1, 0, v_0)^T$, $c_B = (b_1, 0, \gamma_0)^T = (31, 0, 0)^T$ a $c_N = b_2 = 20$. Podľa postupu z časti 1.4 dosadíme pôvodnú účelovú funkciu a doriešime druhú fázu simplexovej metódy.

pravá strana	w_1	w_2	ω	u_1	u_2	v_0
$\frac{ln(5)+10}{2}$	$-\frac{1}{2}$	0	$\frac{ln(5)+10}{2}$	1	$\frac{1}{2}$	0
1	0	0	1	0	0	1
$\frac{-2ln(3)+7ln(5)+2}{2}$	$-\frac{7}{2}$	1	$\frac{-2ln(3)+7ln(5)+2}{2}$	0	$-\frac{3}{2}$	0
$31\frac{ln(5)+10}{2}$	$-31\frac{1}{2}$	0	$31\frac{ln(5)+10}{2}$	0	$-\frac{9}{2}$	0

Získali sme optimálnu tabuľku a tým sme skončili druhú fázu a zároveň prvú iteráciu metódy sečnicových nadrovín. Nové optimálne riešenie je $x^1 = (\frac{31}{2}, 0)^T$. Prejdeme k druhej iterácii, v ktorej najprv pridáme do tabuľky nový stĺpec a takto rozšírenú tabuľku riešime.

2.iterácia.

Vyrátame

$$g_1 = \nabla f(x^1) = \begin{pmatrix} \frac{\partial f(x^1)}{\partial x_1^1} \\ \frac{\partial f(x^1)}{\partial x_2^1} \end{pmatrix} = \begin{pmatrix} 1 + ln(x_1^1) + p_1 \\ 0 + p_2 \end{pmatrix} = \begin{pmatrix} ln(\frac{31}{10}) - 9 \\ -ln(3) - 34 \end{pmatrix}$$

$$\gamma_1 = \nabla f(x^1)x^1 - f(x^1) = g_1^T x^1 - x_1^1 ln(x_1^1) - x_2^1 ln(x_2^1) - p^T x^1 = \frac{31}{2}$$

Stĺpec $v_1 = (-g_1, 1, -\gamma_1)^T$ prenásobime rozšírenou inverznou bázou (ktorú tvoria stĺpce premenných w, ω z konečnej tabuľky prvej iterácie a stĺpec premennej z).

$$\begin{array}{c} \begin{matrix} w_1 & w_2 & \omega & z \end{matrix} \\ \left(\begin{array}{ccc|c} -\frac{1}{2} & 0 & \frac{ln(5)+10}{2} & 0 \\ 0 & 0 & 1 & 0 \\ -\frac{7}{2} & 1 & \frac{-2ln(3)+7ln(5)+2}{2} & 0 \\ \hline -\frac{31}{2} & 0 & \frac{31}{2}(ln(5)+10) & 1 \end{array}\right) \end{array} * \begin{array}{c} v_1 \\ \left(\begin{array}{c} -ln(\frac{31}{10}) + 9 \\ ln(3) + 34 \\ 1 \\ \hline -\frac{31}{2} \end{array}\right) \end{array} = \begin{array}{c} v_1 \\ \left(\begin{array}{c} \frac{1}{2}(ln(\frac{31}{2}) + 1) \\ 1 \\ \frac{7}{2}(ln(\frac{31}{2}) + 1) \\ \hline \frac{31}{2} ln(\frac{31}{2}) \end{array}\right) \end{array}$$

Upravený stĺpec v_1 pridáme do simplexovej tabuľky z konca prvej iterácie. Pokračujeme riešením rozšírenej simplexovej tabuľky.

pravá strana	w_1	w_2	ω	u_1	u_2	v_0	v_1
$\frac{ln(5)+10}{2}$	$-\frac{1}{2}$	0	$\frac{ln(5)+10}{2}$	1	$\frac{1}{2}$	0	$\frac{1}{2}(ln(\frac{31}{2})+1)$
1	0	0	1	0	0	1	1
$\frac{-2ln(3)+7ln(5)+2}{2}$	$-\frac{7}{2}$	1	$\frac{-2ln(3)+7ln(5)+2}{2}$	0	$-\frac{3}{2}$	0	$\frac{7}{2}(ln(\frac{31}{2})+1)$
$31\frac{ln(5)+10}{2}$	$-31\frac{1}{2}$	0	$31\frac{ln(5)+10}{2}$	0	$-\frac{9}{2}$	0	$\frac{31}{2}ln(\frac{31}{2})$

Za pivota volíme prvok v stĺpci v_1 v treťom riadku.

pravá strana	w_1	w_2	ω
$\frac{ln(3)+34}{7}$	0	$-\frac{1}{7}$	$\frac{ln(3)+34}{7}$
$\frac{7ln(\frac{31}{10})+2ln(3)+5}{7(ln(\frac{31}{2})+1)}$	$\frac{1}{ln(\frac{31}{2})+1}$	$\frac{-2}{7(ln(\frac{31}{2})+1)}$	$\frac{7ln(\frac{31}{10})+2ln(3)+5}{7(ln(\frac{31}{2})+1)}$
$\frac{-2ln(3)+7ln(5)+2}{7(ln(\frac{31}{2})+1)}$	$\frac{-1}{ln(\frac{31}{2})+1}$	$\frac{2}{7(ln(\frac{31}{2})+1)}$	$\frac{-2ln(3)+7ln(5)+2}{7(ln(\frac{31}{2})+1)}$
$\frac{62ln(\frac{31}{2})(ln(3)+34)+31*7(ln(5)+10)}{2*7(ln(\frac{31}{2})+1)}$	$\frac{31}{2}\frac{-1}{ln(\frac{31}{2})+1}$	$\frac{-31}{7}\frac{ln(\frac{31}{2})}{ln(\frac{31}{2})+1}$	$\frac{62ln(\frac{31}{2})(ln(3)+34)+31*7(ln(5)+10)}{2*7(ln(\frac{31}{2})+1)}$

u_1	u_2	v_0	v_1
1	$\frac{5}{7}$	0	0
0	$\frac{3}{7(ln(\frac{31}{2})+1)}$	1	0
0	$\frac{-3}{7(ln(\frac{31}{2})+1)}$	0	1
0	$\frac{3}{2}\frac{10ln(\frac{31}{2})-21}{7(ln(\frac{31}{2})+1)}$	0	0

Za pivota volíme prvok v stĺpci u_2 v druhom riadku.

pravá strana	w_1	w_2	ω	u_1	u_2	v_0	v_1
$\frac{-5ln(\frac{31}{10})-ln(3)+11}{3}$	$-\frac{5}{3}$	$\frac{1}{3}$	$\frac{-5ln(\frac{31}{10})-ln(3)+11}{3}$	1	0	$\frac{-5(ln(\frac{31}{2})+1)}{3}$	0
$\frac{7ln(\frac{31}{10})+2ln(3)+5}{3}$	$\frac{7}{3}$	$-\frac{2}{3}$	$\frac{7ln(\frac{31}{10})+2ln(3)+5}{3}$	0	1	$\frac{7(ln(\frac{31}{2})+1)}{3}$	0
1	0	0	1	0	0	1	1
$\frac{3(ln(3)+34)+5(ln(5)+10)}{1}$ $\frac{-5(ln(\frac{31}{2})+1)+\frac{31}{2}}{1}$	-5	-3	$\frac{3(ln(3)+34)+5(ln(5)+10)}{1}$ $\frac{-5(ln(\frac{31}{2})+1)+\frac{31}{2}}{1}$	0	0	$\frac{-10ln(\frac{31}{2})+21}{2}$	0

Získali sme optimálnu tabuľku a skončili druhú iteráciu. Nové optimálne riešenie je $x^2 = (5,3)^T$. Prejdeme na tretiu iteráciu, v ktorej opakujeme postup z druhej iterácie.

3.iterácia.

Vyrátame

$$g_2 = \nabla f(x^2) = \begin{pmatrix} \frac{\partial f(x^2)}{\partial x_1^2} \\ \frac{\partial f(x^2)}{\partial x_2^2} \end{pmatrix} = \begin{pmatrix} 1 + ln(x_1^2) + p_1 \\ 1 + ln(x_2^2) + p_2 \end{pmatrix} = \begin{pmatrix} -9 \\ -33 \end{pmatrix}$$

$$\gamma_2 = \nabla f(x^2)x^2 - f(x^2) = g_2^T x^2 - x_1^2 ln(x_1^2) - x_2^2 ln(x_2^2) - p^T x^2 = 8$$

Do tabuľky pridáme upravený stĺpec $v_2 = (-g_2, 1, -\gamma_2)^T$

$$\left(\begin{array}{cc|c|c} w_1 & w_2 & \omega & z \\ -\frac{5}{3} & \frac{1}{3} & \frac{-5ln(\frac{31}{10})-ln(3)+11}{3} & 0 \\ \frac{7}{3} & -\frac{2}{3} & \frac{7ln(\frac{31}{10})+2ln(3)+5}{3} & 0 \\ 0 & 0 & 1 & 0 \\ \hline -5 & -3 & \frac{3(ln(3)+34)+5(ln(5)+10)}{1} & 1 \\ & & \frac{-5(ln(\frac{31}{2})+1)+\frac{31}{2}}{1} & \end{array}\right) * \left(\begin{array}{c} v_2 \\ 9 \\ 33 \\ 1 \\ \hline -8 \end{array}\right) = \left(\begin{array}{c} v_2 \\ \frac{-1-5ln(\frac{31}{10})-ln(3)}{3} \\ \frac{2+7ln(\frac{31}{10})+2ln(3)}{3} \\ 1 \\ \hline \frac{3(ln(3)+34)+5(ln(5)+10)}{1} \\ \frac{-5(ln(\frac{31}{2})+1)+\frac{31}{2}}{1} - 152 \end{array}\right)$$

Upravený stĺpec v_2 pridáme do tabuľky z konca druhej iterácie a riešime rozšírenú simplexovú tabuľku. Poznamenáme, že v ďalšom výpočte neuvádzame stĺpec ω, lebo predstavuje nie podstatnú informáciu pri simplexových iteráciách. Zároveň je rovnaký ako stĺpec pravej strany a teda nestrácame o ňom informáciu.

pravá strana	w_1	w_2	u_1	u_2	v_0	v_1	v_2
$\frac{-5ln(\frac{31}{10})-ln(3)+11}{3}$	$-\frac{5}{3}$	$\frac{1}{3}$	1	0	$\frac{-5(ln(\frac{31}{2})+1)}{3}$	0	$\frac{-1-5ln(\frac{31}{10})-ln(3)}{3}$
$\frac{7ln(\frac{31}{10})+2ln(3)+5}{3}$	$\frac{7}{3}$	$-\frac{2}{3}$	0	1	$\frac{7(ln(\frac{31}{2})+1)}{3}$	0	$\frac{2+7ln(\frac{31}{10})+2ln(3)}{3}$
1	0	0	0	0	1	1	1
$\frac{3(ln(3)+34)+5(ln(5)+10)}{1}$	-5	-3	0	0	$\frac{-10ln(\frac{31}{2})+21}{2}$	0	$\frac{3(ln(3)+34)+5(ln(5)+10)}{1}$
$\frac{-5(ln(\frac{31}{2})+1)+\frac{31}{2}}{1}$							$\frac{-5(ln(\frac{31}{2})+1)+\frac{31}{2}}{1}-152$

Za pivota volíme prvok v stĺpci v_2 v treťom riadku.

pravá strana	w_1	w_2	u_1	u_2	v_0	v_1	v_2
4	$-\frac{5}{3}$	$\frac{1}{3}$	1	0	$\frac{-5ln(5)+ln(3)-4}{3}$	$\frac{1+5ln(\frac{31}{10})+ln(3)}{3}$	0
1	$\frac{7}{3}$	$-\frac{2}{3}$	0	1	$\frac{7ln(5)-2ln(3)+5}{3}$	$\frac{-2-7ln(\frac{31}{10})-2ln(3)}{3}$	0
1	0	0	0	0	1	1	1
152	-5	-3	0	0	$\frac{-3(ln(3)+34)-5(ln(5)+10)}{1}$	$\frac{-3(ln(3)+34)-5(ln(5)+10)}{1}$	0
					$+152$	$\frac{+5(ln(\frac{31}{2})+1)-\frac{31}{2}}{1}+152$	

Získali sme optimálne riešenie $x^3 = (5,3)^T$, $z_3 = -152 = f(x^3)$. Keďže $f(x^3) = f(\hat{x})$ a $x^3 = \hat{x}$, tretia iterácia určila presné optimálne riešenie. V ďalších iteráciách už nemožno pokračovať, pretože príslušné γ_3 po prenásobení rovné nule indikuje, že x^3 je optimálnym riešením aj v novej rozšírenej simplexovej tabuľke.

4 Numerické experimenty

4.1 Popis testujúcich úloh

V numerických experimentoch budeme skúmať metódu sečnicových nadrovín na špeciálnej úlohe (1)

$$Min\{f(x)|\ Ax \leq b,\ x \geq 0\} \qquad \textbf{(1*)}$$

kde matica $A \in R^{m\times n}$, vektor $b \in R^m$, $b \geq 0$ a účelová funkcia $f(x)$ pozostáva z lineárnej formy $p^T x$ (pomocou ktorej regulujeme polohu optimálneho riešenia $\hat{x}$) a nelineárnej funckie $F(x)$.
Teda $f(x) = F(x) + p^T x$, kde za $F(x)$ volíme jednu z nasledovných funkcií

$$F_1(x) = x^T Cx \quad (C = C^T\ kladne\ definitna) \tag{13}$$

$$F_2(x) = \frac{1}{4\omega}(x^T Dx)^2 + x^T Cx \quad (D > 0\ \ diagonalna, \quad \omega = \|D\|_F^2 = (\textstyle\sum_{i=1}^m \sum_{j=1}^n |d_{ij}|^2)) \tag{14}$$

$$F_3(x) = \sum_{j=1}^{n} x_j ln(x_j) \qquad (pre\ x_i = 0,\ x_i ln(x_i) = 0) \tag{15}$$

$$F_4(x) = \sum_{j=1}^{n} e^{r_j x_j + s_j} \qquad (0 \leq r_j \leq 1,\ 0 \leq s_j \leq 1) \tag{16}$$

4.2 Generovanie testujúcich úloh so známym optimálnym riešením

V časti Základné pojmy (str. 10) sme definovali Lagrangeovu funkciu a Kuhn-Tuckerove podmienky. Pre našu úlohu (1*) Lagrangeova funkcia má tvar

$$L(x,u) = F(x) + p^T x + u^T(Ax - b)$$

a Kuhn Tuckerove podmienky $(K-T)$ pre úlohu (1*) nadobúdajú tvar

$$\begin{array}{rcl@{\qquad}rcl} \nabla F(\hat{x}) + p + A^T\hat{u} & \geq & 0 & A\hat{x} - b & \leq & 0 \\ \hat{x}^T(\nabla F(\hat{x}) + p + A^T\hat{u}) & = & 0 & \hat{u}^T(A\hat{x} - b) & = & 0 \\ \hat{x} & \geq & 0 & \hat{u} & \geq & 0 \end{array}$$

Uvedené vzťahy nám umožňujú pre danú funkciu $F(x)$ a zvolené optimálne riešenie $\hat{x} \geq 0$ dodefinovať ostatné veličiny $\hat{u}$, A, b, p podľa nasledovnej výpočtovej schémy

Generátor úlohy (1*)

I) Zvolíme rozmer úlohy:
$n =$ počet premenných
$m =$ počet ohraničení

II) Zvolíme maticu $A \in R^{m \times n} \quad (A = (a_{ij}) > 0)$

III) Zvolíme optimálne riešenie $\hat{x} \in R^n_+$ a príslušný vektor Lagrangeových multiplikátorov $\hat{u} \in R^m_+$

IV) Zvolíme vektory $\hat{v} \in R^n_+$ a $\hat{y} \in R^m_+$ tak aby

$$\begin{array}{llll} \hat{x}_j + \hat{v}_j > 0 & a & \hat{x}_j\, \hat{v}_j = 0\,, & j = 1, \ldots, n \\ \hat{u}_i + \hat{y}_i > 0 & a & \hat{u}_i\, \hat{y}_i = 0\,, & i = 1, \ldots, m \end{array}$$

V) Vypočítame $\nabla F(\hat{x})$

VI) Vypočítame vektory p a b
$p = \hat{v} - A^T \hat{u} - \nabla F(\hat{x})$
$b = \hat{y} + A\hat{x}$

4.3 Popis experimentov

Na začiatku vyberieme účelovú funkciu z (13), (14), (15), (16), ktorú budeme testovať. K nej máme naprogramované dve funkcie

i) FunkcnaHodnota(n,x) - vypočíta hodnotu účelovej funkcie

ii) Gradient(n,x) - vypočíta gradient účelovej funkcie

Zvolíme rozmer n, m. Spustíme Generátor úloh, ktorý určí základné parametre $\hat{x}$, $\hat{u}$, A, b, p a špeciálné parametre pre (13) C, pre (14) C, D, ω, pre (16) r, s.

Potom spustíme program, ktorý našu úlohu vyrieši s približným riešením x^k, $f(x^k)$ a dolným odhadom $z^k \leq f(x^k)$. Mnohonásobným opakovaním (10 krát, resp. 5 krát) získame priemerné údaje, na ktorých sledujeme

i) vplyv rozmerov úlohy na počet iterácií

ii) vplyv pomeru m/n na normovaný počet iterácií

iii) vplyv absolútnej presnosti na počet iterácií

iv) vplyv počtu iterácií na presnosť

Po vyriešení série úloh výstupné informácie zapisujeme do tabuliek. Riadky tabuliek obsahujú informáciu o rozmere série úloh, teda jeden riadok tabuľky predstavuje úlohy s rovnakým počtom premenných n a počtom ohraničení m. Stĺpce tabuliek obsahujú nasledovné výstupné informácie

a) Počet iterácii metódy sečnicových nadrovín (značíme MSN)

b) Počet iterácii simplexovej metódy

c) Čas výpočtu

d) Absolútna presnosť riešenia $\varepsilon = \|\hat{x} - x^k\|$

e) Relatívna presnosť riešenia $\frac{\|\hat{x}-x^k\|}{1+\|\hat{x}\|}$

f) Vzdialenosť optimálnej hodnoty účelovej funkcie od dolného odhadu $f(\hat{x}) - z^k$ (v tabuľkách značíme $\hat{f} - z^k$) ($z = \xi$ v úlohe (4) a pre z^k platí vzťah (5))

g) Vzdialenosť priebežnej hodnoty účelovej funkcie od optimálnej hodnoty účelovej funkcie $f(x^k) - f(\hat{x})$ (v tabuľkách značíme $f_k - \hat{f}$)

4.4 Výsledky Experimentu - Vplyv rozmerov úlohy na počet iterácií

Budeme sledovať vplyv rozmerov úlohy (1*) na počet iterácií metódy sečnicových nadrovín. Pre každú účelovú funkciu (13), (14), (15) a (16) sme testovanie rozdelili do dvoch častí, kde v prvej časti sledujeme zväčšovanie počtu premenných n vzhľadom na tri úrovne počtu ohraničení m a v druhej časti sledujeme zväčšovanie počtu ohraničení m vzhľadom na tri úrovne počtu premenných n. Výsledky sme na základe toho rozdelili do dvoch tabuliek.

Charakteristika tabuliek. Jeden riadok tabuľky predstavuje sadu 10 resp. 5 riešených úloh s rovnakým rozmerom, pričom do tabuľky sme zapísali priemerné výsledky z výstupných informácií. V prvej tabuľke sú úlohy s postupným zväčšovaním n od 2 do 95 pre $m = 10, 50, 100$ a v druhej tabuľke sú úlohy s postupným zväčšovaním m od 6 do 150 pre $n = 5, 50, 100$.
Pozn. stopovým kritériom je dosiahnutie presnosti $\hat{f} - z^k = 10^{-12}$

4.4.1 Kvadratická funkcia

$f(x) = x^T Cx + p^T x$

n	m	priemer iterácií MSN	priemerný počet simplexových iterácii	čas (mm:ss)	$\left\|\hat{x} - x^k\right\|$	$\frac{\left\|x^k - \hat{x}\right\|}{1 + \left\|\hat{x}\right\|}$	$\hat{f} - z^k$	$f_k - \hat{f}$
2	10	2,2	4,7	00:00	1,31E-18	1,34E-19	0,00E+00	0,00E+00
3	10	2,9	8,0	00:00	3,38E-16	2,71E-17	0,00E+00	0,00E+00
5	10	7,7	26,4	00:00	1,14E-17	1,02E-18	0,00E+00	0,00E+00
8	10	17,8	90,8	00:00	5,65E-17	3,44E-18	0,00E+00	0,00E+00
10	50	4,6	57,6	00:00	8,64E-17	4,66E-18	0,00E+00	0,00E+00
25	50	30,9	1153,7	00:00	1,14E-15	4,05E-17	0,00E+00	0,00E+00
38	50	87,9	4545,3	00:03	4,04E-15	1,20E-16	0,00E+00	0,00E+00
45	50	136,8	7644,3	00:08	1,31E-14	3,54E-16	0,00E+00	0,00E+00
40	100	32,9	3126,8	00:02	2,79E-15	8,02E-17	0,00E+00	0,00E+00
50	100	53,5	6625,2	00:09	4,80E-15	1,25E-16	0,00E+00	0,00E+00
75	100	165,0	28625,8	01:32	2,29E-14	4,88E-16	0,00E+00	0,00E+00
95	100	378,0	66297,8	07:49	1,21E-13	2,28E-15	0,00E+00	0,00E+00

Tabuľka 1: Úlohy s kvadratickou účelovou funkciou, s postupným zväčšovaním počtu premenných n od 2 do 95 a tromi úrovňami počtu ohraničení $m = 10, 50$ a 100

Komentár k Tabuľke 1. Pre pevne stanovené m so zväčšovaním n sa zvyšuje počet iterácii a čas výpočtu a znižuje sa presnosť riešenia. Pri zväčšení rozmeru úlohy z $n = 38$ a $m = 50$ na nový rozmer $n = 45$ a $m = 50$ priemerný počet iterácii metódy sečnicových nadrovín vzrástol o asi 50 iterácií, čas výpočtu sa zväčšil o 5 sekúnd a presnosť sa zhoršila o veľmi malú hodnotu. Keď sme zväčšili rozmer z $n = 45$ a $m = 50$ na nový rozmer $n = 50$ a $m = 100$, potom priemerný počet iterácii metódy sečnicových nadrovín sa zmenšil o asi 80 iterácií, čo asi spôsobuje veľký nárast počtu ohraničení. Pre úlohy s najväčším rozmerom $n = 95$ a $m = 100$ bola dosiahnutá veľmi dobrá presnosť riešenia a čas výpočtu bol $7 : 49$.

n	m	priemer iterácií MSN	priemerný počet simplexových iterácii	čas (mm:ss)	$\left\|\hat{x} - x^k\right\|$	$\frac{\left\|x^k - \hat{x}\right\|}{1 + \left\|\hat{x}\right\|}$	$\hat{f} - z^k$	$f_k - \hat{f}$
5	6	11,2	35,7	00:00	2,60E-17	1,88E-18	0,00E+00	0,00E+00
5	8	6,8	25,8	00:00	1,16E-17	8,87E-19	0,00E+00	0,00E+00
5	12	4,7	19,8	00:00	1,46E-17	9,75E-19	0,00E+00	0,00E+00
5	20	2,2	9,6	00:00	8,36E-18	7,18E-19	0,00E+00	0,00E+00
50	55	166,2	10009,3	00:13	1,73E-14	2,91E-16	0,00E+00	0,00E+00
50	60	134,0	9944,5	00:12	9,57E-15	2,46E-16	0,00E+00	0,00E+00
50	70	96,5	8710,8	00:10	6,46E-15	1,67E-16	0,00E+00	0,00E+00
50	100	50,3	6256,1	00:08	4,87E-15	1,32E-16	0,00E+00	0,00E+00
100	110	318,6	73251,0	10:49	9,08E-14	1,70E-15	0,00E+00	0,00E+00
100	150	174,2	58406,0	09:16	6,27E-14	1,13E-15	0,00E+00	0,00E+00

Tabuľka 2: Úlohy s kvadratickou účelovou funkciou, s postupným zväčšovaním počtu ohraničení m od 6 do 150 a tromi úrovňami počtu premenných $n = 5, 50$ a 100

Komentár k Tabuľke 2. Pre pevne stanovené n so zvyšovaním m sa znižuje počet iterácii a zväčšujú sa hodnoty presnosti. Keď zväčšíme rozmer úlohy z $n = 50$ a $m = 55$ na nový rozmer $n = 50$ a $m = 60$ priemerný

počet iterácií metódy sečnicových nadrovín klesne o 30 iterácii, čas výpočtu sa zmenšil o 1 sekundu a presnosť sa zlepšila o malú hodnotu. Pre úlohy s najväčším počtom premenných $n = 100$ a ohraničení $m = 110$ bola dosiahnutá hodnota času výpočtu $10 : 49$. Keď sme do úloh s $n = 100$ a $m = 110$ pridali 40 ohraničení, tak pre nové úlohy s rozmerom $n = 100$ a $m = 150$ bol čas výpočtu o asi 36 percent menší a dosiahol hodnotu $6 : 51$.

4.4.2 Bikvadratická funkcia

$f(x) = \frac{1}{4\omega}(x^T Dx)^2 + x^T Cx + p^T x$

n	m	priemer iterácií MSN	priemerný počet simplexových iterácii	čas (mm:ss)	$\left\|\hat{x} - x^k\right\|$	$\frac{\left\|x^k - \hat{x}\right\|}{1 + \left\|\hat{x}\right\|}$	$\hat{f} - z^k$	$f_k - \hat{f}$
2	10	2,0	4,5	00:00	5,36E-19	6,85E-20	0,00E+00	0,00E+00
3	10	2,9	8,4	00:00	1,69E-18	1,92E-19	0,00E+00	0,00E+00
5	10	8,0	32,1	00:00	2,03E-17	1,88E-18	0,00E+00	0,00E+00
8	10	16,7	82,9	00:00	1,65E-16	1,05E-17	0,00E+00	0,00E+00
10	50	4,0	42,1	00:00	1,43E-16	7,78E-18	0,00E+00	0,00E+00
25	50	31,1	1205,6	00:00	9,06E-16	3,27E-17	0,00E+00	0,00E+00
38	50	95,7	4907,3	00:05	1,12E-14	3,30E-16	0,00E+00	0,00E+00
45	50	148,8	8380,7	00:12	1,68E-14	4,56E-16	0,00E+00	0,00E+00
40	100	34,2	3323,8	00:03	3,36E-15	9,33E-17	0,00E+00	0,00E+00
50	100	57,9	7054,8	00:12	5,46E-15	1,38E-16	0,00E+00	0,00E+00
75	100	166,4	29572,2	01:59	2,87E-14	6,30E-16	0,00E+00	0,00E+00
95	100	366,0	68669,6	10:14	9,61E-14	1,79E-15	0,00E+00	0,00E+00

Tabuľka 3: Úlohy s bikvadratickou účelovou funkciou, s postupným zväčšovaním počtu premenných n od 2 do 95 a tromi úrovňami počtu ohraničení $m = 10, 50$ a 100

Komentár k Tabuľke 3. Pre pevne stanovené m so zväčšovaním n sa zvyšuje počet iterácii a čas výpočtu a znižuje sa presnosť riešenia. Pri zväčšení rozmeru úlohy z $n = 38$ a $m = 50$ na nový rozmer $n = 45$ a $m = 50$ priemerný počet iterácii metódy sečnicových nadrovín vzrastol o asi 50 iterácií, čas výpočtu sa zväčšil o 7 sekúnd a presnosť sa zhoršila o nepatrnú hodnotu, čo sú podobné výsledky ako pre kvadratickú účelovú funckiu. Keď

sme zväčšili rozmer z $n = 45$ a $m = 50$ na nový rozmer $n = 50$ a $m = 100$, potom priemerný počet iterácii metódy sečnicových nadrovín sa zmenšil o asi 90 iterácií, čo asi spôsobuje veľký nárast počtu ohraničení. Pre úlohy s najväčším rozmerom $n = 95$ a $m = 100$ bola dosiahnutá veľmi dobrá presnosť riešenia a čas výpočtu bol $10 : 14$.

n	m	priemer iterácií MSN	priemerný počet simplexových iterácii	čas (mm:ss)	$\left\|\hat{x} - x^k\right\|$	$\frac{\left\|x^k - \hat{x}\right\|}{1 + \left\|\hat{x}\right\|}$	$\hat{f} - z^k$	$f_k - \hat{f}$
5	6	11,7	37,5	00:00	3,52E-16	2,22E-17	0,00E+00	0,00E+00
5	8	8,4	30,1	00:00	1,57E-16	1,11E-17	0,00E+00	0,00E+00
5	12	5,8	26,2	00:00	1,93E-17	1,51E-18	0,00E+00	0,00E+00
5	20	3,5	18,2	00:00	6,79E-17	4,95E-18	0,00E+00	0,00E+00
50	55	166,4	11022,6	00:20	1,85E-14	4,88E-16	0,00E+00	0,00E+00
50	60	137,5	10415,0	00:17	1,08E-14	2,76E-16	0,00E+00	0,00E+00
50	70	102,5	9221,6	00:14	5,58E-15	1,45E-16	0,00E+00	0,00E+00
50	100	55,4	6722,5	00:10	5,42E-15	1,37E-16	0,00E+00	0,00E+00
100	110	368,8	82943,0	15:21	1,23E-13	2,21E-15	0,00E+00	0,00E+00
100	150	157,6	54953,2	09:06	4,70E-14	8,85E-16	0,00E+00	0,00E+00

Tabuľka 4: Úlohy s bikvadratickou účelovou funkciou, s postupným zväčšovaním počtu ohraničení m od 6 do 150 a tromi úrovňami počtu premenných $n = 5, 50$ a 100

Komentár k Tabuľke 4. Pre pevne stanovené n so zvyšovaním m sa znižuje počet iterácii a zväčšujú sa hodnoty presnosti. Keď zväčšíme rozmer úlohy z $n = 50$ a $m = 55$ na nový rozmer $n = 50$ a $m = 60$ priemerný počet iterácií metódy sečnicových nadrovín klesne o asi 30 iterácii, čas výpočtu sa zmenšil o 3 sekundu a presnosť sa zlepšila o malú hodnotu, čo sú podobné výsledky ako pre kvadratickú funkciu. Pre úlohy s najväčším počtom premenných $n = 100$ a ohraničení $m = 110$ bola dosiahnutá pomerne vysoká hodnota času výpočtu $15 : 21$ a v porovnaní s kvadratickou účelovou funkciou trval čas výpočtu viac asi o 8 minút. Keď sme do úloh s $n = 100$ a $m = 110$ pridali 40 ohraničení, tak pre nové úlohy s rozmerom $n = 100$ a $m = 150$ bol čas výpočtu o asi 40 percent menší a dosiahol hodnotu $9 : 06$.

4.4.3 Logaritmická funkcia

$f(x) = \sum_{j=1}^{n} x_j ln(x_j) + p^T x$

n	m	priemer iterácií MSN	priemerný počet simplexových iterácii	čas (mm:ss)	$\left\|\hat{x} - x^k\right\|$	$\frac{\left\|x^k - \hat{x}\right\|}{1 + \left\|\hat{x}\right\|}$	$\hat{f} - z^k$	$f_k - \hat{f}$
2	10	2,0	4,1	00:00	5,41E-18	8,97E-19	0,00E+00	0,00E+00
3	10	2,0	5,0	00:00	3,91E-18	3,87E-19	0,00E+00	0,00E+00
5	10	2,0	7,6	00:00	9,73E-17	7,84E-18	0,00E+00	0,00E+00
8	10	3,9	13,3	00:00	5,42E-08	3,65E-09	0,00E+00	0,00E+00
10	50	2,0	15,3	00:00	1,78E-16	9,28E-18	0,00E+00	0,00E+00
25	50	2,0	37,3	00:00	6,94E-15	2,55E-16	0,00E+00	0,00E+00
38	50	2,0	55,5	00:00	3,63E-15	1,04E-16	0,00E+00	0,00E+00
45	50	2,6	71,4	00:00	7,02E-15	1,91E-16	0,00E+00	0,00E+00
40	100	2,0	68,7	00:00	1,35E-15	3,90E-17	0,00E+00	0,00E+00
50	100	2,0	81,2	00:00	3,64E-15	9,06E-17	0,00E+00	0,00E+00
75	100	2,0	150,6	00:00	1,56E-14	3,14E-16	0,00E+00	0,00E+00
95	100	3,6	171,2	00:00	6,03E-08	1,10E-09	0,00E+00	0,00E+00

Tabuľka 5: Úlohy s logaritmickou účelovou funkciou, s postupným zväčšovaním počtu premenných n od 2 do 95 a tromi úrovňami počtu ohraničení $m = 10, 50$ a 100

Komentár k Tabuľke 5. Úlohy s logaritmickou účelovou funkciou boli vo väčšine prípadoch vyrátané len s dvoma iteráciami metódy sečnicových nadrovín. Len pri špeciálnych typoch úloh s rozmermi, v ktorých bol počet premenných n blízky počtu ohraničení m bolo vykonaných viac ako 2 iterácie metódy sečnicových nadrovín. Ďalej pozorujeme, že so zväčšovaním rozmeru úloh sa zvyšuje počet simplexových iterácii a zmenšuje sa presnosť riešenia. Zaujímavé sú nulové hodnoty časov výpočtu ("čo znamená, že výpočet bol ukončený ihneď po stlačení tlačidla počítaj") Pre úlohy s najväčším rozmerom $n = 95$ a $m = 100$ bola dosiahnutá dobrá presnosť riešenia a minimálny čas výpočtu, pričom pre kvadratickú a bikvadratickú účelovú funkciu bol čas výpočtu približne od 7 do 10 minút.

n	m	priemer iterácií MSN	priemerný počet simplexových iterácii	čas (mm:ss)	$\left\|\hat{x}-x^k\right\|$	$\frac{\left\|x^k-\hat{x}\right\|}{1+\left\|\hat{x}\right\|}$	$\hat{f}-z^k$	$f_k-\hat{f}$
5	6	4,2	9,6	00:00	1,62E-08	1,37E-09	0,00E+00	0,00E+00
5	8	2,0	8,0	00:00	4,43E-17	3,17E-18	0,00E+00	0,00E+00
5	12	2,0	7,6	00:00	1,49E-17	1,22E-18	0,00E+00	0,00E+00
5	20	2,0	7,7	00:00	3,65E-17	2,91E-18	0,00E+00	0,00E+00
50	55	6,5	83,4	00:00	1,75E-07	4,10E-09	0,00E+00	0,00E+00
50	60	2,0	77,6	00:00	1,18E-14	2,96E-16	0,00E+00	0,00E+00
50	70	2,0	84,0	00:00	1,63E-14	4,29E-16	0,00E+00	0,00E+00
50	100	2,0	86,0	00:00	1,65E-14	4,09E-16	0,00E+00	0,00E+00
100	110	2,4	182,0	00:00	1,67E-13	3,15E-15	0,00E+00	0,00E+00
100	150	2,0	212,0	00:01	5,81E-15	1,08E-16	0,00E+00	0,00E+00

Tabuľka 6: Úlohy s logaritmickou účelovou funkciou, s postupným zväčšovaním počtu ohraničení m od 6 do 150 a tromi úrovňami počtu premenných $n = 5, 50$ a 100

Komentár k Tabuľke 6. Keď zväčšíme rozmer úlohy z $n = 50$ a $m = 55$ na nový rozmer $n = 50$ a $m = 60$ priemerný počet iterácií metódy sečnicových nadrovín klesne o $4,5$ iterácii na 2 iterácie. Pri ďalšom zväčšovaní m je počet iterácii MSN rovnaký na úrovni 2 iterácii. Pre pevne stanovené n so zvyšovaním m sa zväčšujú hodnoty presnosti. Pre úlohy s najväčším rozmerom $n = 100$,$m = 110$ a $n = 100$, $m = 150$ bola dosiahnutá veľmi dobrá presnosť, pre úlohy s väčším počtom ohraničení $m = 150$ lepšia ako pre úlohy s menším počtom ohraničení $m = 110$. Pre úlohy s najväčším rozmerom $n = 100$, $m = 110$ a $n = 100$, $m = 150$ bol počet simplexových iterácií približne 200, čo je veľký rozdiel od kvadratickej a bikvadratickej účelovej funkcie, kde počet simplexových iterácii bol približne $300-$krát väčší a dosahoval hodnoty okolo 60000 iterácii.

4.4.4 Exponenciálna funkcia

$f(x) = \sum_{j=1}^{n} e^{r_j x_j + s_j} + p^T x$

n	m	priemer iterácií MSN	priemerný počet simplexových iterácii	čas (mm:ss)	$\left\|\hat{x} - x^k\right\|$	$\frac{\left\|x^k - \hat{x}\right\|}{1 + \left\|\hat{x}\right\|}$	$\hat{f} - z^k$	$f_k - \hat{f}$
2	10	2,3	5,0	00:00	7,37E-19	9,90E-20	0,00E+00	0,00E+00
3	10	3,1	7,8	00:00	4,14E-18	4,36E-19	0,00E+00	0,00E+00
5	10	9,2	22,7	00:00	1,59E-16	1,10E-17	0,00E+00	0,00E+00
8	10	15,2	46,0	00:00	1,84E-16	1,10E-17	0,00E+00	0,00E+00
10	50	8,4	36,8	00:00	1,14E-15	7,39E-17	0,00E+00	0,00E+00
25	50	51,9	419,8	00:00	1,70E-15	5,86E-17	0,00E+00	0,00E+00
38	50	87,1	814,4	00:01	3,30E-08	9,44E-10	-4,71E-05	4,71E-05
45	50	91,8	836,5	00:01	1,66E-08	4,45E-10	4,66E-07	-4,66E-07
40	100	51,4	574,1	00:00	9,45E-15	2,71E-16	0,00E+00	0,00E+00
50	100	67,0	943,5	00:02	7,94E-15	1,98E-16	0,00E+00	0,00E+00
75	100	91,4	1813,2	00:06	2,31E-12	4,80E-14	-6,05E-10	6,05E-10
95	100	112,4	2522,0	00:26	6,29E-14	1,23E-15	0,00E+00	0,00E+00

Tabuľka 7: Úlohy s exponencialnou účelovou funkciou, s postupným zväčšovaním počtu premenných n od 2 do 95 a tromi úrovňami počtu ohraničení $m = 10, 50$ a 100

Komentár k Tabuľke 7. Pre pevne stanovené m so zväčšovaním n sa zvyšuje počet iterácii a čas výpočtu a znižuje sa presnosť riešenia. V porovnaní s kvadratickou a bikvadratickou funkciou sú nárasty iterácii menšie. Napríklad pri zväčšení rozmeru úlohy z $n = 38$ a $m = 50$ na nový rozmer $n = 45$ a $m = 50$ priemerný počet iterácii metódy sečnicových nadrovín sa zväčšil o približne 5 iterácií, zatiaľ čo v rovnakom prípade rozmerov úloh u kvadratickej a bikvadratickej funkcie bol nárast o 50 iterácií. Keď sme zväčšili rozmer z $n = 45$ a $m = 50$ na nový rozmer $n = 50$ a $m = 100$, potom priemerný počet iterácii metódy sečnicových nadrovín sa zmenšil o asi 25 iterácií, čo asi spôsobuje veľký nárast počtu ohraničení. Pre úlohy s najväčším rozmerom $n = 95$ a $m = 100$ bola dosiahnutá veľmi dobrá presnosť riešenia. Nevýhodou exponenciálnej funkcie je vysoká náročnosť výpočtu,

keďže v niektorých príkladoch vznikali veľké čísla radu 10^{38}. Výsledkom sú niektoré nepresné hodnoty v tabuľke pre $n = 38$ a $m = 50$, $n = 45$ a $m = 50$, $n = 75$ a $m = 100$ a nespoľahlivosť, pretože v niektorých prípadoch počítanie zlyhalo.

n	m	priemer iterácií MSN	priemerný počet simplexových iterácii	čas (mm:ss)	$\left\|\hat{x}-x^k\right\|$	$\frac{\left\|x^k-\hat{x}\right\|}{1+\left\|\hat{x}\right\|}$	$\hat{f}-z^k$	$f_k-\hat{f}$
5	6	9,9	20,2	00:00	2,00E-07	1,52E-08	0,00E+00	0,00E+00
5	8	9,6	25,4	00:00	5,35E-16	3,04E-17	0,00E+00	0,00E+00
5	12	5,4	18,8	00:00	3,25E-14	2,26E-15	0,00E+00	0,00E+00
5	20	3,5	11,5	00:00	1,07E-17	8,21E-19	0,00E+00	0,00E+00
50	55	139,7	1654,0	00:03	2,74E-04	7,35E-06	8,09E-02	-8,09E-02
50	60	107,6	1338,3	00:02	8,37E-15	2,19E-16	0,00E+00	0,00E+00
50	70	93,7	1194,3	00:02	1,94E-04	4,28E-06	1,09E-01	-1,09E-01
50	100	37,4	694,6	00:01	2,95E-15	7,98E-17	0,00E+00	0,00E+00
100	110	163,4	3334,4	00:27	3,82E-14	7,34E-16	0,00E+00	0,00E+00
100	150	171,8	3775,0	00:48	8,66E-14	1,67E-15	0,00E+00	0,00E+00

Tabuľka 8: Úlohy s exponencialnou účelovou funkciou, s postupným zväčšovaním počtu ohraničení m od 6 do 150 a tromi úrovňami počtu premenných $n = 5, 50$ a 100

Komentár k Tabuľke 8. Pre pevne stanovené n so zvyšovaním m sa znižuje počet iterácii a zväčšujú sa hodnoty presnosti. Pri zväčšení rozmeru úlohy z $n = 50$ a $m = 55$ na nový rozmer $n = 50$ a $m = 60$ priemerný počet iterácií metódy sečnicových nadrovín klesol o približne 30 iterácii, čas výpočtu sa zmenšil o 1 sekundu a presnosť sa zlepšila, čo sú podobné výsledky ako pre kvadratickú a bikvadratickú funkciu. Pre úlohy s najväčším počtom premenných $n = 100$ a ohraničení $m = 110$ bolo vykonaných približne rovnaký počet iterácií metódy sečnicových nadrovín ako pre kvadratickú a bikvadratickú funkciu, ale s podstatne menším počtom simplexových iterácií.

4.5 Výsledky Experimentu - Vplyv pomeru m/n na normovaný počet iterácií

Zdá sa byť zaujímavé odsledovať vzťah medzi počtom premenných, počtom ohraničení a počtom iterácii metódy sečnicových nadrovín. Pri riešení úloh s rôznym počtom premenných n a ohraničení m v predchádzajúcom experimente sme si všimli, že so zväčšovaním počtu ohraničení m vzhľadom na rovnaký počet premenných n je vykonaných menej iterácii metódy sečnicových nadrovín.

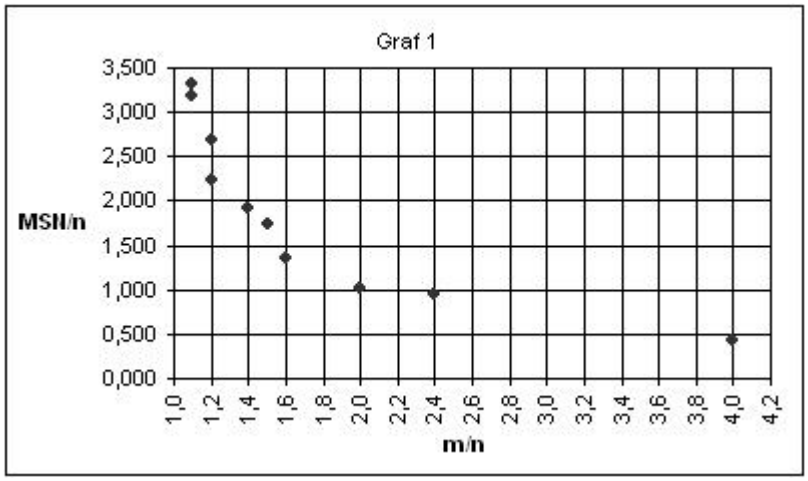
Charakteristika tabuliek. Do tabuliek zapíšeme počet iterácií metódy sečnicových nadrovín potrebných na dosiahnutie žiadanej presnosti. Použijeme výstupy z tabuliek z predchádzajúcej časti, v ktorých skúmame úlohy s postupným zväčšovaním počtu ohraničení m od 6 do 150 a tromi úrovňami počtu premenných $n = 5, 50, 100$. Ďalej pridáme stĺpec $\frac{m}{n}$, ktorý predstavuje podiel počtu ohraničení m a počtu premenných n, stĺpec $\frac{MSN}{n}$ je podiel priemerného počtu iterácii metódy sečnicových nadrovín a počtu premenných n. Pre lepší prehľad sme údaje z tabuliek preniesli do grafov.

4.5.1 Kvadratická funkcia

$f(x) = x^T Cx + p^T x$

n	m	priemer iterácií MSN	$\frac{m}{n}$	$\frac{MSN}{n}$
5	6	11,2	1,2	2,240
5	8	6,8	1,6	1,360
5	12	4,7	2,4	0,940
5	20	2,2	4,0	0,440
50	55	166,2	1,1	3,324
50	60	134,0	1,2	2,680
50	70	96,5	1,4	1,930
50	100	50,3	2,0	1,006
100	110	318,6	1,1	3,186
100	150	174,2	1,5	1,742

Tabuľka 9: Úlohy s kvadratickou účelovou funkciou, s postupným zväčšovaním počtu ohraničení m od 6 do 150 a tromi úrovňami počtu premenných $n = 5, 50$ a 100



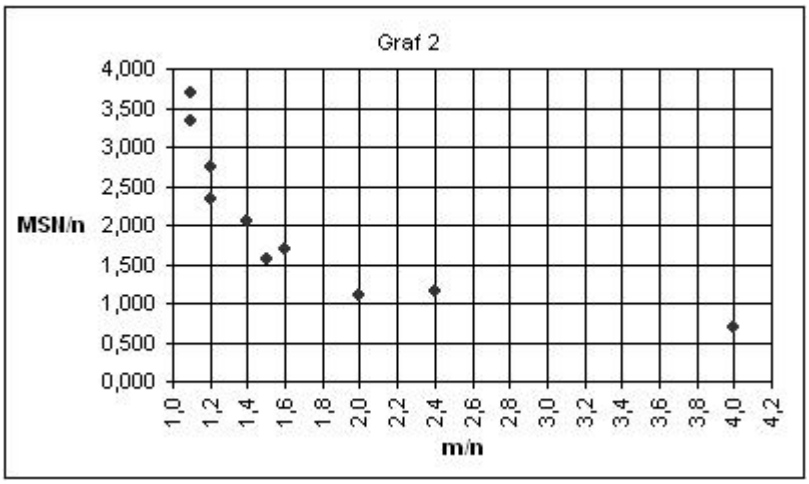
Graf 1: Údaje z Tabuľky 9

4.5.2 Bikvadratická funkcia

$f(x) = \frac{1}{4\omega}(x^T Dx)^2 + x^T Cx + p^T x$

n	m	priemer iterácií MSN	$\frac{m}{n}$	$\frac{MSN}{n}$
5	6	11,7	1,2	2,340
5	8	8,4	1,6	1,680
5	12	5,8	2,4	1,160
5	20	3,5	4,0	0,700
50	55	166,4	1,1	3,328
50	60	137,5	1,2	2,750
50	70	102,5	1,4	2,050
50	100	55,4	2,0	1,108
100	110	368,8	1,1	3,688
100	150	157,6	1,5	1,576

Tabuľka 10: Úlohy s bikvadratickou účelovou funkciou, s postupným zväčšovaním počtu ohraničení m od 6 do 150 a tromi úrovňami počtu premenných $n = 5, 50$ a 100



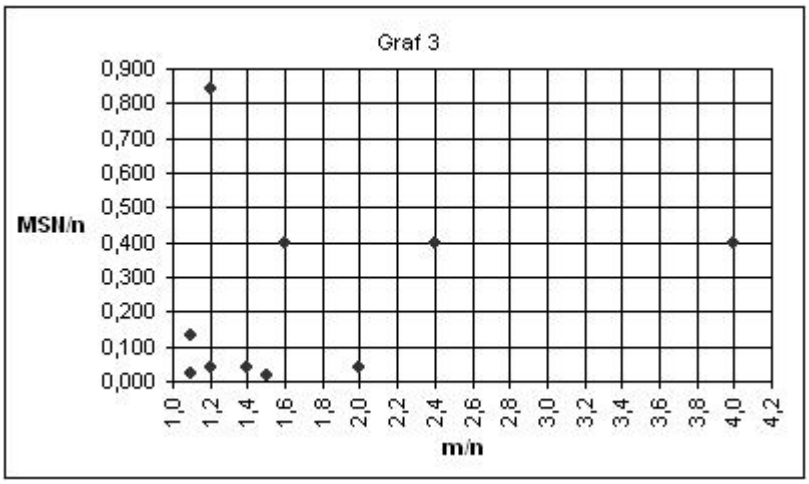
Graf 2: Údaje z Tabuľky 10

4.5.3 Logaritmická funkcia

$f(x) = \sum_{j=1}^{n} x_j ln(x_j) + p^T x$

n	m	priemer iterácií MSN	$\frac{m}{n}$	$\frac{MSN}{n}$
5	6	4,2	1,2	0,840
5	8	2,0	1,6	0,400
5	12	2,0	2,4	0,400
5	20	2,0	4,0	0,400
50	55	6,5	1,1	0,130
50	60	2,0	1,2	0,040
50	70	2,0	1,4	0,040
50	100	2,0	2,0	0,040
100	110	2,4	1,1	0,024
100	150	2,0	1,5	0,020

Tabuľka 11: Úlohy s logaritmickou účelovou funkciou, s postupným zväčšovaním počtu ohraničení m od 6 do 150 a tromi úrovňami počtu premenných $n = 5, 50$ a 100



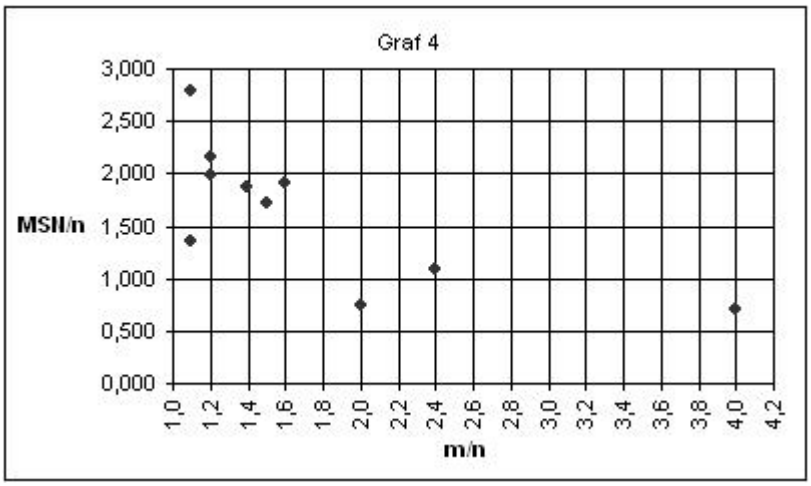
Graf 3: Údaje z Tabuľky 11

4.5.4 Exponenciálna funkcia

$f(x) = \sum_{j=1}^{n} e^{r_j x_j + s_j} + p^T x$

n	m	priemer iterácií MSN	$\frac{m}{n}$	$\frac{MSN}{n}$
5	6	9,9	1,2	1,980
5	8	9,6	1,6	1,920
5	12	5,4	2,4	1,080
5	20	3,5	4,0	0,700
50	55	139,7	1,1	2,794
50	60	107,6	1,2	2,152
50	70	93,7	1,4	1,874
50	100	37,4	2,0	0,748
100	110	135,0	1,1	1,350
100	150	171,8	1,5	1,718

Tabuľka 12: Úlohy s exponenciálnou účelovou funkciou, s postupným zväčšovaním počtu ohraničení m od 6 do 150 a tromi úrovňami počtu premenných $n = 5, 50$ a 100



Graf 3: Údaje z Tabuľky 12

Komentár k Tabuľkám 9, 10, 11, 12. Pre kvadratickú, bikvadratickú a exponenciálnu účelovú funkciu bol pri rovnakom rozmere úlohy podobný počet iterácií metódy sečnicových nadrovín, pričom pri rovnakom počte premenných a so zväčšujúcim počtom ohraničení klesá počet iterácií metódy sečnicových nadrovín. Pre tieto účelové funkcie ak je počet ohraničení m väčší ako počet premenných $1,2-$krát je počet iterácií metódy sečnicových nadrovín približne $2,1$ až $2,7$ -krát väčší ako počet premenných (pre úlohu s $n = 50$ a $m = 60$ je urobených približne 105 až 135 iterácií MSN). Pre úlohy s logaritmickou účelovou funkciou vo väčšine prípadoch postačili 2 iterácie metódy sečnicových nadrovín, s výnimkou úloh, v ktorých bol počet ohraničení m približne rovnaký ako počet premenných n (pozri tab. 11 úlohy $n = 5$ a $m = 6$, $n = 50$ a $m = 55$, $n = 100$ a $m = 110$).

4.6 Výsledky Experimentu - Vplyv absolútnej presnosti na počet iterácií

V experimente budeme sledovať vplyv vopred zadanej absolútnej presnosti $\varepsilon = \|\hat{x} - x^k\|$ na počet iterácii metódy sečnicových nadrovín. Teda stanovíme presnosť $\varepsilon > 0$ a pozorujeme koľko iterácii je potrebných na získanie tejto presnosti, pričom opäť budeme riešiť sady úloh s rôznym počtom premenných n a ohraničení m pre každú účelových funkciu z (13), (14), (15) a (16).

Charakteristika tabuliek. Jeden riadok tabuľky predstavuje sadu 5 riešených úloh s rovnakým rozmerom, pričom do tabuľky sme zapísali priemerné hodnoty z výstupných informácií. Stanovíme tri hodnoty požadovanej presnosti $\varepsilon_1 = 10$, $\varepsilon_2 = 0,1$ a $\varepsilon_3 = 0,001$ ktoré budú našim stopovým kritériom. Výsledky zapíšeme do troch tabuliek pre každú účelovú funkciu.

4.6.1 Kvadratická funkcia

$f(x) = x^T C x + p^T x$

n	m	priemer iterácií MSN	priemerný počet simplexových iterácii	čas (mm:ss)	$\left\|\hat{x} - x^k\right\|$	$\frac{\left\|x^k - \hat{x}\right\|}{1 + \left\|\hat{x}\right\|}$	$\hat{f} - z^k$	$f_k - \hat{f}$
5	6	1,2	4,6	00:00	6,82E+00	5,23E-01	2,23E+04	2,49E+03
5	8	2,0	9,8	00:00	7,63E+00	5,70E-01	1,36E+04	3,31E+03
10	11	7,6	67,0	00:00	7,01E+00	3,74E-01	1,29E+04	6,05E+03
10	16	3,8	43,6	00:00	6,82E+00	4,23E-01	2,33E+04	5,31E+03
25	30	19,8	664,4	00:00	8,07E+00	3,09E-01	4,87E+04	1,44E+04
25	40	17,2	736,8	00:00	8,20E+00	2,85E-01	3,88E+04	2,01E+04
50	55	70,4	6597,2	00:08	8,72E+00	2,35E-01	5,69E+04	3,24E+04
50	100	36,0	5052,4	00:08	8,29E+00	2,16E-01	9,51E+04	4,40E+04
75	80	138,6	25541,8	01:32	8,22E+00	1,69E-01	1,02E+05	4,35E+04
75	100	96,4	21442,4	01:15	9,09E+00	1,97E-01	1,01E+05	6,37E+04
100	110	190,4	60702,6	06:09	9,36E+00	1,71E-01	1,60E+05	7,87E+04
100	150	110,2	45145,6	05:14	8,17E+00	1,53E-01	1,78E+05	7,15E+04

Tabuľka 13: Úlohy s kvadratickou účelovou funkciou, s postupným zväčšovaním počtu premenných n od 5 do 100 a počtu ohraničení

m od 6 do 150 a požadovanou presnosťou $\varepsilon \leq \varepsilon_1 = 10$

n	m	priemer iterácií MSN	priemerný počet simplexových iterácii	čas (mm:ss)	$\left\|\hat{x} - x^k\right\|$	$\frac{\left\|x^k - \hat{x}\right\|}{1 + \left\|\hat{x}\right\|}$	$\hat{f} - z^k$	$f_k - \hat{f}$
5	6	9,8	35,2	00:00	3,13E-02	2,19E-03	4,63E+01	5,69E-01
5	8	6,2	26,8	00:00	3,23E-02	2,36E-03	4,19E+01	5,35E-01
10	11	27,0	170,0	00:00	1,17E-02	6,16E-04	2,30E+00	2,94E-02
10	16	15,0	125,2	00:00	3,57E-02	1,74E-03	8,05E+00	1,50E+00
25	30	57,6	1421,2	00:00	5,67E-02	2,20E-03	1,90E+01	2,36E+00
25	40	26,4	1115,0	00:00	4,54E-04	1,72E-05	6,52E+01	9,45E-02
50	55	174,8	11362,4	00:16	6,60E-02	1,64E-03	1,32E+01	2,57E+00
50	100	54,6	6805,6	00:10	6,09E-03	1,46E-04	1,56E+02	2,63E+00
75	80	248,4	33748,8	01:53	7,61E-02	1,57E-03	4,04E+01	5,84E+00
75	100	160,2	28801,8	01:40	3,84E-02	8,32E-04	1,97E+01	5,09E+00
100	110	325,0	72441,8	10:15	7,24E-02	1,35E-03	3,91E+01	7,65E+00
100	150	160,6	54333,8	07:43	1,54E-02	2,77E-04	1,11E+02	4,08E+00

Tabuľka 14: Úlohy s kvadratickou účelovou funkciou, s postupným zväčšovaním počtu premenných n od 5 do 100 a počtu ohraničení m od 6 do 150 a požadovanou presnosťou $\varepsilon \leq \varepsilon_2 = 0,1$

n	m	priemer iterácií MSN	priemerný počet simplexových iterácii	čas (mm:ss)	$\left\|\hat{x} - x^k\right\|$	$\frac{\left\|x^k - \hat{x}\right\|}{1 + \left\|\hat{x}\right\|}$	$\hat{f} - z^k$	$f_k - \hat{f}$
5	6	9,8	29,4	00:00	2,53E-04	2,25E-05	7,84E+00	2,35E-03
5	8	6,8	24,0	00:00	1,07E-17	8,76E-19	1,39E+03	0,00E+00
10	11	26,2	142,2	00:00	3,50E-04	1,94E-05	7,83E-01	7,18E-04
10	16	19,0	162,6	00:00	1,14E-16	5,91E-18	6,27E+00	0,00E+00
25	30	59,6	1402,8	00:00	2,31E-15	8,48E-17	8,12E-02	0,00E+00
25	40	31,8	1238,6	00:00	1,58E-15	5,10E-17	2,01E+01	0,00E+00
50	55	163,4	10738,4	00:15	9,66E-15	2,38E-16	3,48E-02	0,00E+00
50	100	54,4	7055,0	00:10	8,79E-15	2,18E-16	4,93E+01	0,00E+00
75	80	278,4	32789,6	02:20	3,49E-04	7,54E-06	1,57E-02	6,90E-04
75	100	173,0	30349,2	02:03	3,13E-14	6,51E-16	2,55E+00	0,00E+00
100	110	331,2	73868,8	09:44	2,38E-04	4,27E-06	2,95E-01	3,84E-03
100	150	171,2	58961,4	07:48	6,67E-14	1,22E-15	1,16E+01	0,00E+00

Tabuľka 15: Úlohy s kvadratickou účelovou funkciou, s postupným zväčšovaním počtu premenných n od 5 do 100 a počtu ohraničení m od 6 do 150 a požadovanou presnosťou $\varepsilon \leq \varepsilon_3 = 0,001$

Komentár k Tabuľám 13, 14, 15. Z tabuliek (13), (14) a (15) vidíme, že s naročnejšími požiadavkami na presnosť riešenia ε sa zväčšuje počet iterácií metódy sečnicových nadrovín a počet simplexových iterácii a následne rastie hodnota času výpočtu. Medzi tabuľkami (13) a (14) sú väššie nárasty ako medzi tabuľkami (14) a (15). Pre úlohy s počtom premenných $n = 5$ a počtom ohraničení $m = 6$ bol priemerný počet iterácii metódy sečnicových nadrovín v tabuľke (13) $MSN = 1,2$ v tabuľkách (14) a (15) rovnaký $MSN = 9,8$. Na dosiahnutie presnosti $\varepsilon_2 = 0,1$ počítač urobil približne rovnaký počet iterácií (MSN aj simplexových) ako na dosiahnutie presnosti $\varepsilon_3 = 0,001$, pričom na úlohách s väčším rozmerom vidíme, že viac iterácií bolo urobených v tabuľke (15) ako v tabuľke (14). I keď rozdiely v počte iterácií medzi tabuľkami (14) a (15) nie sú veľké, v poslednej tabuľke (15) bola dosiahnutá lepšia presnosť riešenia. Všimnime si, že v poslednej tabuľke (15) je veľmi dobrá presnosť funkčnej hodnoty $f(x^k)$, zatiaľ čo dolný odhad funkčnej hodnoty z^k je nepresný (v prvom experimente sme získali veľmi dobrý dolný odhad z^k).

4.6.2 Bikvadratická funkcia

$f(x) = \frac{1}{4\omega}(x^T Dx)^2 + x^T Cx + p^T x$

n	m	priemer iterácií MSN	priemerný počet simplexových iterácii	čas (mm:ss)	$\left\|\hat{x} - x^k\right\|$	$\frac{\left\|x^k - \hat{x}\right\|}{1 + \left\|\hat{x}\right\|}$	$\hat{f} - z^k$	$f_k - \hat{f}$
5	6	2,2	9,2	00:00	7,59E+00	5,62E-01	1,68E+04	4,83E+03
5	8	2,0	10,4	00:00	5,20E+00	4,04E-01	7,54E+03	2,15E+03
10	11	6,6	46,4	00:00	6,41E+00	3,61E-01	2,05E+04	5,85E+03
10	16	3,8	36,0	00:00	6,76E+00	3,97E-01	2,51E+04	6,72E+03
25	30	23,8	838,8	00:00	8,86E+00	3,24E-01	5,13E+04	2,08E+04
25	40	12,4	586,8	00:00	7,38E+00	2,80E-01	4,98E+04	1,80E+04
50	55	74,6	7377,2	00:09	8,72E+00	2,32E-01	6,61E+04	3,27E+04
50	100	36,6	5520,0	00:08	7,71E+00	1,95E-01	1,08E+05	3,44E+04
75	80	141,2	26907,8	01:54	9,23E+00	1,96E-01	1,04E+05	5,59E+04
75	100	96,4	21945,0	01:26	8,83E+00	1,88E-01	1,38E+05	6,16E+04
100	110	198,8	62851,6	08:49	8,90E+00	1,64E-01	1,19E+05	7,17E+04
100	150	112,4	46804,6	05:23	8,60E+00	1,61E-01	1,69E+05	6,53E+04

Tabuľka 16: Úlohy s bikvadratickou účelovou funkciou, s postupným zväčšovaním počtu premenných n od 5 do 100 a počtu ohraničení m od 6 do 150 a požadovanou presnosťou $\varepsilon \leq \varepsilon_1 = 10$

n	m	priemer iterácií MSN	priemerný počet simplexových iterácii	čas (mm:ss)	$\left\|\hat{x} - x^k\right\|$	$\frac{\left\|x^k - \hat{x}\right\|}{1 + \left\|\hat{x}\right\|}$	$\hat{f} - z^k$	$f_k - \hat{f}$
5	6	8,8	27,8	00:00	3,36E-02	2,36E-03	7,98E+00	2,16E-01
5	8	9,8	34,4	00:00	9,24E-03	7,14E-04	8,26E+00	2,03E-01
10	11	25,4	175,2	00:00	3,83E-02	2,13E-03	1,28E+01	5,12E-01
10	16	17,2	150,8	00:00	1,16E-16	6,36E-18	4,29E+00	0,00E+00
25	30	67,0	1671,4	00:00	4,65E-02	1,66E-03	2,94E+01	1,63E+00
25	40	28,6	1090,8	00:00	6,50E-16	2,32E-17	1,09E+01	0,00E+00
50	55	143,6	9879,2	00:16	7,14E-02	1,89E-03	1,31E+01	4,21E+00
50	100	46,8	6090,6	00:09	2,77E-02	7,45E-04	1,40E+02	1,38E+01
75	80	276,6	36456,6	02:38	6,28E-02	1,34E-03	1,45E+01	3,87E+00
75	100	177,4	31953,0	02:05	3,13E-02	6,49E-04	2,05E+01	3,15E+00
100	110	338,2	75816,2	12:00	5,24E-02	9,88E-04	1,34E+01	4,81E+00
100	150	168,0	58112,4	08:04	2,35E-02	4,50E-04	9,88E+01	9,81E+00

Tabuľka 17: Úlohy s bikvadratickou účelovou funkciou, s postupným zväčšovaním počtu premenných n od 5 do 100 a počtu ohraničení m od 6 do 150 a požadovanou presnosťou $\varepsilon \leq \varepsilon_2 = 0,1$

n	m	priemer iterácií MSN	priemerný počet simplexových iterácii	čas (mm:ss)	$\left\|\hat{x} - x^k\right\|$	$\frac{\left\|x^k - \hat{x}\right\|}{1 + \left\|\hat{x}\right\|}$	$\hat{f} - z^k$	$f_k - \hat{f}$
5	6	7,6	25,8	00:00	1,40E-17	1,17E-18	2,26E+00	0,00E+00
5	8	9,2	34,4	00:00	8,55E-18	6,58E-19	3,43E+00	0,00E+00
10	11	39,4	197,8	00:00	5,08E-04	2,74E-05	3,80E-01	4,25E-05
10	16	14,4	137,8	00:00	9,01E-17	5,85E-18	7,55E+00	0,00E+00
25	30	68,8	1734,0	00:00	1,30E-15	4,47E-17	6,73E-01	0,00E+00
25	40	29,2	1158,2	00:00	8,23E-16	3,08E-17	2,40E+01	0,00E+00
50	55	161,2	10888,8	00:19	2,17E-04	5,48E-06	6,87E-02	1,62E-03
50	100	49,4	6406,4	00:10	5,16E-15	1,37E-16	1,18E+01	0,00E+00
75	80	296,4	36696,0	02:43	3,27E-04	6,76E-06	4,60E-02	3,79E-04
75	100	169,2	30067,8	02:05	3,91E-14	8,40E-16	4,72E+00	0,00E+00
100	110	312,6	72608,8	12:19	1,27E-13	2,39E-15	3,08E-01	0,00E+00
100	150	177,6	61076,2	09:48	6,44E-14	1,15E-15	1,01E+01	0,00E+00

Tabuľka 18: Úlohy s bikvadratickou účelovou funkciou, s postupným zväčšovaním počtu premenných n od 5 do 100 a počtu ohraničení m od 6 do 150 a požadovanou presnosťou $\varepsilon \leq \varepsilon_3 = 0,001$

Komentár k Tabuľám 16, 17, 18. Z tabuliek (16), (17) a (18) vidíme, že s naročnejšími požiadavkami na presnosť riešenia ε sa zväčšuje počet iterácií metódy sečnicových nadrovín a počet simplexových iterácii a následne rastie hodnota času výpočtu. Medzi tabuľkami (16) a (17) sú väššie nárasty ako medzi tabuľkami (17) a (18). Pre úlohy s počtom premenných $n = 5$ a počtom ohraničení $m = 6$ bol priemerný počet iterácii metódy sečnicových nadrovín v tabuľke (16) $MSN = 2,2$ v tabuľkách (17) a (18) približne $MSN = 8$, čo sú podobné výsledky ako pre kvadratickú účelovú funkciu. Podobne ako pre kvadratickú na dosiahnutie presnosti $\varepsilon_2 = 0,1$ počítač urobil približne rovnaký počet iterácií (MSN aj simplexových) ako na dosiahnutie presnosti $\varepsilon_3 = 0,001$, pričom na úlohách s väčším rozmerom vidíme, že bolo urobených viac iterácií v tabuľke (18) ako v tabuľke (17).

I keď rozdiely v počte iterácií medzi tabuľkami (17) a (18) nie sú veľké, v poslednej tabuľke (18) bola dosiahnutá lepšia presnosť riešenia. Tak ako pre kvadratickú funkciu, aj v tomto prípade v poslednej tabuľke (18) pre $\varepsilon_3 = 0,001$ sme získali veľmi dobrú presnosť funkčnej hodnoty $f(x^k)$, zatiaľ čo dolný odhad funkčnej hodnoty z^k je nepresný (v prvom experimente sme získali veľmi dobrý dolný odhad z^k).

4.6.3 Logaritmická funkcia

$f(x) = \sum_{j=1}^{n} x_j ln(x_j) + p^T x$

n	m	priemer iterácií MSN	priemerný počet simplexových iterácii	čas (mm:ss)	$\left\|\hat{x} - x^k\right\|$	$\frac{\left\|x^k - \hat{x}\right\|}{1+\left\|\hat{x}\right\|}$	$\hat{f} - z^k$	$f_k - \hat{f}$
5	6	1,0	6,0	00:00	3,42E-17	2,75E-18	3,50E+01	0,00E+00
5	8	1,0	6,4	00:00	1,93E-17	1,21E-18	4,56E+01	0,00E+00
10	11	1,0	12,4	00:00	5,92E-02	3,52E-03	6,69E+01	4,97E-03
10	16	1,0	13,4	00:00	1,54E-16	7,31E-18	1,06E+02	0,00E+00
25	30	1,0	32,0	00:00	3,03E-15	1,25E-16	1,80E+02	0,00E+00
25	40	1,0	37,0	00:00	5,82E-16	1,97E-17	2,20E+02	0,00E+00
50	55	1,0	78,8	00:00	2,85E-02	6,96E-04	4,30E+02	4,08E-06
50	100	1,0	92,0	00:00	2,37E-15	6,31E-17	3,79E+02	0,00E+00
75	80	1,0	119,2	00:00	5,72E-15	1,22E-16	5,90E+02	0,00E+00
75	100	1,0	139,4	00:00	3,35E-14	6,62E-16	6,32E+02	0,00E+00
100	110	1,0	162,4	00:00	7,68E-15	1,45E-16	7,58E+02	0,00E+00
100	150	1,0	200,0	00:00	3,52E-15	6,50E-17	8,20E+02	0,00E+00

Tabuľka 19: Úlohy s logaritmickou účelovou funkciou, s postupným zväčšovaním počtu premenných n od 5 do 100 a počtu ohraničení m od 6 do 150 a požadovanou presnosťou $\varepsilon \leq \varepsilon_1 = 10$

n	m	priemer iterácií MSN	priemerný počet simplexových iterácii	čas (mm:ss)	$\left\|\hat{x}-x^k\right\|$	$\frac{\left\|x^k-\hat{x}\right\|}{1+\left\|\hat{x}\right\|}$	$\hat{f}-z^k$	$f_k-\hat{f}$
5	6	1,0	6,2	00:00	1,74E-17	1,38E-18	3,62E+01	0,00E+00
5	8	1,0	6,2	00:00	1,42E-17	1,05E-18	4,37E+01	0,00E+00
10	11	1,0	11,8	00:00	5,06E-17	3,20E-18	6,26E+01	0,00E+00
10	16	1,0	14,8	00:00	4,78E-16	2,44E-17	8,31E+01	0,00E+00
25	30	1,0	33,4	00:00	1,69E-15	6,01E-17	2,02E+02	0,00E+00
25	40	1,0	37,0	00:00	4,24E-15	1,51E-16	1,86E+02	0,00E+00
50	55	1,0	72,4	00:00	5,76E-15	1,44E-16	3,94E+02	0,00E+00
50	100	1,0	86,8	00:00	4,73E-14	1,31E-15	3,97E+02	0,00E+00
75	80	1,4	112,8	00:00	1,15E-02	2,36E-04	4,74E+02	8,44E-05
75	100	1,0	137,4	00:00	7,26E-15	1,41E-16	6,20E+02	0,00E+00
100	110	1,0	161,8	00:00	2,64E-14	4,81E-16	8,10E+02	0,00E+00
100	150	1,0	208,8	00:00	1,48E-13	2,82E-15	7,83E+02	0,00E+00

Tabuľka 20: Úlohy s logaritmickou účelovou funkciou, s postupným zväčšovaním počtu premenných n od 5 do 100 a počtu ohraničení m od 6 do 150 a požadovanou presnosťou $\varepsilon \leq \varepsilon_2 = 0,1$

n	m	priemer iterácií MSN	priemerný počet simplexových iterácii	čas (mm:ss)	$\left\|\hat{x}-x^k\right\|$	$\frac{\left\|x^k-\hat{x}\right\|}{1+\left\|\hat{x}\right\|}$	$\hat{f}-z^k$	$f_k-\hat{f}$
5	6	1,0	5,8	00:00	2,40E-17	1,72E-18	4,24E+01	0,00E+00
5	8	1,0	6,4	00:00	2,29E-17	1,85E-18	4,00E+01	0,00E+00
10	11	6,2	19,0	00:00	1,80E-04	1,05E-05	6,06E+01	2,26E-08
10	16	1,0	13,8	00:00	1,24E-16	6,10E-18	9,19E+01	0,00E+00
25	30	1,0	33,8	00:00	7,43E-15	2,59E-16	2,05E+02	0,00E+00
25	40	1,0	37,6	00:00	1,84E-14	6,58E-16	1,89E+02	0,00E+00
50	55	1,0	74,0	00:00	8,75E-15	2,19E-16	4,23E+02	0,00E+00
50	100	1,0	83,6	00:00	2,00E-15	5,13E-17	3,76E+02	0,00E+00
75	80	1,0	121,6	00:00	3,87E-14	7,80E-16	5,74E+02	0,00E+00
75	100	1,0	134,0	00:00	2,45E-15	5,13E-17	5,95E+02	0,00E+00
100	110	1,0	170,2	00:00	2,46E-14	4,44E-16	7,89E+02	0,00E+00
100	150	1,0	190,2	00:00	2,54E-14	4,90E-16	8,13E+02	0,00E+00

Tabuľka 21: Úlohy s logaritmickou účelovou funkciou, s postupným zväčšovaním počtu premenných n od 5 do 100 a počtu ohraničení

m od 6 do 150 a požadovanou presnosťou $\varepsilon \leq \varepsilon_3 = 0,001$

Komentár k Tabuľám 19, 20, 21. Pre úlohy s logaritmickou účelovou funkciou vo väčšine prípadoch bola potrebná jedna iterácia metódy sečnicových nadrovín na všetky požadované presnosti, výnimku tvoria len úlohy, v ktorých počet premenných je blízky počtu ohraničení ($n = 10, m = 11$ a $n = 75, m = 80$). So zväčšovaním rozmeru úloh rastie počet simplexových iterácií a znižujú sa hodnoty presnosti riešenia. Pre úlohy s najväčším rozmerom bol počet simplexových iterácií asi 200, čo je približne 300− krát menej ako pre kvadratickú a bikvadratickú funkciu. V poslednej tabuľke (20) s požadovanou presnosťou $\varepsilon_3 = 0,001$ sme získali veľmi dobré hodnoty pre presnosti (podobné výsledky ako v prvom experimente), s výnimkou dolného odhad funkčnej hodnoty z_k, ktorý je nepresný.

4.6.4 Exponenciálna funkcia

$f(x) = \sum_{j=1}^{n} e^{r_j x_j + s_j} + p^T x$

n	m	priemer iterácií MSN	priemerný počet simplexových iterácii	čas (mm:ss)	$\left\|\hat{x} - x^k\right\|$	$\frac{\left\|x^k - \hat{x}\right\|}{1+\left\|\hat{x}\right\|}$	$\hat{f} - z^k$	$f_k - \hat{f}$
5	6	2,0	7,6	00:00	5,53E+00	4,51E-01	1,15E+03	2,62E+04
5	8	2,6	7,2	00:00	1,68E+00	1,19E-01	3,01E+03	2,41E+04
10	11	7,6	18,8	00:00	6,73E+00	3,76E-01	5,06E+03	1,32E+05
10	16	8,0	20,6	00:00	6,40E+00	3,36E-01	8,58E+03	6,87E+04
25	30	25,6	155,2	00:00	4,39E+00	1,59E-01	5,48E+03	1,90E+04
25	40	44,6	235,6	00:00	7,31E+00	2,38E-01	2,20E+04	1,96E+05
50	55	88,0	870,8	00:01	7,99E+00	2,17E-01	8,95E+03	8,50E+03
50	100	31,4	413,0	00:00	6,83E+00	1,76E-01	9,76E+03	2,75E+04
75	80	141,8	1561,6	00:06	7,37E+00	1,59E-01	1,12E+04	5,75E+03
75	100	91,8	1004,6	00:04	8,79E+00	1,90E-01	1,55E+04	3,90E+04
100	110	103,8	2703,4	00:16	8,46E+00	1,63E-01	1,36E+04	9,26E+03
100	150	115,2	3103,6	00:19	8,11E+00	1,58E-01	1,47E+04	1,11E+04

Tabuľka 22: Úlohy s exponenciálnou účelovou funkciou, s postupným zväčšovaním počtu premenných n od 5 do 100 a počtu ohraničení m od 6 do 150 a požadovanou presnosťou $\varepsilon \leq \varepsilon_1 = 10$

n	m	priemer iterácií MSN	priemerný počet simplexových iterácii	čas (mm:ss)	$\left\|\hat{x}-x^k\right\|$	$\frac{\left\|x^k-\hat{x}\right\|}{1+\left\|\hat{x}\right\|}$	$\hat{f}-z^k$	$f_k-\hat{f}$
5	6	8,2	19,2	00:00	4,82E-02	3,40E-03	4,24E+00	1,39E+00
5	8	5,4	14,0	00:00	2,19E-02	1,82E-03	2,08E+02	7,47E-01
10	11	24,0	67,8	00:00	2,97E-02	1,43E-03	6,29E+01	1,16E+00
10	16	6,4	32,0	00:00	2,03E-16	1,07E-17	8,29E+01	0,00E+00
25	30	57,0	342,6	00:00	4,58E-02	1,58E-03	7,09E+00	6,52E-01
25	40	33,2	331,6	00:00	4,85E-03	1,63E-04	1,08E+02	1,24E+00
50	55	133,8	1322,2	00:03	3,43E-02	8,86E-04	5,13E+00	1,02E+00
50	100	51,8	829,6	00:02	1,31E-14	3,69E-16	4,02E+02	0,00E+00
75	80	147,8	2191,6	00:07	3,67E-02	8,02E-04	6,16E+00	3,46E-01
75	100	113,2	1717,4	00:07	2,72E-02	6,18E-04	2,56E+01	2,15E+00
100	110	70,0	2387,0	00:11	6,76E-02	1,32E-03	9,43E+00	1,11E+00
100	150	43,6	1853,8	00:10	5,70E-15	1,05E-16	2,31E+01	0,00E+00

Tabuľka 23: Úlohy s exponenciálnou účelovou funkciou, s postupným zväčšovaním počtu premenných n od 5 do 100 a počtu ohraničení m od 6 do 150 a požadovanou presnosťou $\varepsilon \leq \varepsilon_2 = 0,1$

n	m	priemer iterácií MSN	priemerný počet simplexových iterácii	čas (mm:ss)	$\left\|\hat{x}-x^k\right\|$	$\frac{\left\|x^k-\hat{x}\right\|}{1+\left\|\hat{x}\right\|}$	$\hat{f}-z^k$	$f_k-\hat{f}$
5	6	11,2	23,6	00:00	7,59E-05	7,39E-06	1,07E-01	1,70E-05
5	8	8,0	27,4	00:00	4,45E-17	4,18E-18	8,06E+01	0,00E+00
10	11	10,2	32,4	00:00	4,99E-05	2,86E-06	2,91E+00	1,86E-08
10	16	18,2	69,6	00:00	8,93E-17	4,89E-18	4,05E+01	0,00E+00
25	30	39,2	142,6	00:00	1,61E-05	5,48E-07	4,10E+01	2,16E-09
25	40	29,6	364,8	00:00	9,26E-16	3,55E-17	1,82E+02	0,00E+00
50	55	133,6	1496,8	00:03	3,96E-04	1,01E-05	3,03E-01	1,88E-03
50	100	31,4	571,8	00:00	7,31E-15	1,75E-16	9,04E+02	0,00E+00
75	80	151,8	2706,8	00:09	1,56E-04	3,30E-06	1,87E+00	9,72E-07
75	100	86,6	1832,0	00:06	3,00E-14	6,33E-16	8,05E+00	0,00E+00
100	110	139,8	3875,8	00:26	5,28E-14	9,90E-16	2,00E+01	0,00E+00
100	150	95,4	2923,0	00:20	4,59E-14	8,47E-16	4,44E+01	0,00E+00

Tabuľka 24: Úlohy s exponenciálnou účelovou funkciou, s postupným zväčšovaním počtu premenných n od 5 do 100 a počtu ohraničení

m od 6 do 150 a požadovanou presnosťou $\varepsilon \leq \varepsilon_3 = 0,001$

Komentár k Tabuľám 22, 23, 24. V tabuľkách (22), (23) a (24) vidíme, že na rozdiel od kvadratickej, bikvadratickej a logaritmickej účelovej funkcie neplatí vo všetkých prípadoch, že s náročnejšími požiadavkami na presnosť sa postupne zväčšuje počet iterácií metódy sečnicových nadrovín. Napríklad pre úlohy s počtom premenných $n = 5$ a počtom ohraničení $m = 6$ bol priemerný počet iterácii metódy sečnicových nadrovín v tab. (22) $MSN = 2$ v tab. (23) $MSN = 8,2$ a v tab. (24) $MSN = 11,2$, a pre úlohy s $n = 10$ a $m = 11$ bol počet iterácii v tab. (22) $MSN = 7,6$ v tab. (23) $MSN = 24$ a v tab. (24) $MSN = 10,2$. Pre exponenciálnu funkciu sa vyskytovala nespoľahlivosť pri riešení úloh. So zväčšujúcim počtom premenných n a ohraničení m rástol počet úloh, ktoré počítač nevyriešil. Čo asi súviselo s veľkou výpočtovou náročnosťou exponenciálnej funkcie, v ktorej vznikali veľké čísla (10^{38}). Dôsledkom toho boli nedoriešenie úlohy a nepresnosti.

4.7 Výsledky Experimentu - Vplyv počtu iterácií na presnosť

V tomto experimente budeme sledovať pri vopred zadanom počte iterácií metódy sečnicových nadrovín riešenie úloh s počtom premenných $n = 55$ a počtom ohraničení $m = 60$. Teda obmedzíme počet iterácií metódy sečnicových nadrovín na konkrétnu hodnotu a zisťujeme akú presnosť dosiahneme pre rôzne typy účelových funkcií.

Charakteristika tabuliek. Každý riadok tabuľky predstavuje sadu 5 riešených úloh s počtom premenných $n = 55$, počtom ohraničení $m = 60$ a s vopred zadaným počtom iterácií metódy sečnicových nadrovín, pričom do tabuľky sme zapísali priemerné hodnoty z výstupných informácií. Pre kvadratickú, bikvadratickú a exponenciálnu účelovú funkciu zadáme počet iterácií metódy sečnicových nadrovín $MSN = 100$, 150, 175, 200 a pre logaritmickú $MSN = 1$, 2, 3.

4.7.1 Kvadratická funkcia

$f(x) = x^T Cx + p^T x$

počet iterácií MSN	priemerný počet simplexových iterácii	čas (mm:ss)	$\left\|\hat{x} - x^k\right\|$	$\frac{\left\|x^k - \hat{x}\right\|}{1 + \left\|\hat{x}\right\|}$	$\hat{f} - z^k$	$f_k - \hat{f}$
100	10484	0:00:18	1,27E+01	3,12E-01	4,38E+04	7,38E+04
150	12272	0:00:24	2,17E+00	5,01E-02	2,42E+03	4,29E+03
175	12706	0:00:26	2,99E-01	6,74E-03	1,50E+02	1,78E+02
200	13636,6	0:00:29	1,95E-01	4,84E-03	4,99E+01	9,21E+01

Tabuľka 25: Úlohy s kvadratickou účelovou funkciou,
s počtom premenných $n = 55$, s počtom ohraničení $m = 60$
a s počtom iterácií metódy sečnicových nadrovín od 100 do 200

4.7.2 Bikvadratická funkcia

$f(x) = \frac{1}{4\omega}(x^T Dx)^2 + x^T Cx + p^T x$

počet iterácií MSN	priemerný počet simplexových iterácii	čas (mm:ss)	$\left\|\hat{x} - x^k\right\|$	$\frac{\left\|x^k - \hat{x}\right\|}{1 + \left\|\hat{x}\right\|}$	$\hat{f} - z^k$	$f_k - \hat{f}$
100	10701,2	00:19	7,72E+00	1,90E-01	3,82E+04	-2,76E+04
150	13372,2	00:27	2,04E+00	4,89E-02	3,36E+03	-2,74E+03
175	13672,8	00:30	6,57E-01	1,57E-02	6,03E+02	-5,05E+02
200	14657,0	00:34	5,83E-13	1,46E-14	2,04E-07	0,00E+00

Tabuľka 26: Úlohy s bikvadratickou účelovou funkciou,
s počtom premenných $n = 55$, s počtom ohraničení $m = 60$
a s počtom iterácií metódy sečnicových nadrovín od 100 do 200

4.7.3 Logaritmická funkcia

$f(x) = \sum_{j=1}^{n} x_j ln(x_j) + p^T x$

počet iterácií MSN	priemerný počet simplexových iterácii	čas (mm:ss)	$\left\|\hat{x} - x^k\right\|$	$\frac{\left\|x^k - \hat{x}\right\|}{1 + \left\|\hat{x}\right\|}$	$\hat{f} - z^k$	$f_k - \hat{f}$
1	83,4	00:00	3,36E-02	9,25E-04	4,49E+02	1,01E-03
2	80,0	00:00	2,07E-14	5,21E-16	0,00E+00	0,00E+00
3	83,0	00:00	1,24E-14	3,35E-16	0,00E+00	0,00E+00

Tabuľka 27: Úlohy s logaritmickou účelovou funkciou,
s počtom premenných $n = 55$, s počtom ohraničení $m = 60$
a s počtom iterácií metódy sečnicových nadrovín od 1 do 3

4.7.4 Exponenciálna funkcia

$f(x) = \sum_{j=1}^{n} e^{r_j x_j + s_j} + p^T x$

počet iterácií MSN	priemerný počet simplexových iterácii	čas (mm:ss)	$\left\|\hat{x} - x^k\right\|$	$\frac{\left\|x^k - \hat{x}\right\|}{1 + \left\|\hat{x}\right\|}$	$\hat{f} - z^k$	$f_k - \hat{f}$
100	457,4	00:00	3,58E+01	8,27E-01	5,48E+04	1,02E+11
150	993,0	00:02	1,83E+01	4,33E-01	3,49E+04	3,27E+09
175	1166,8	00:02	1,94E+01	4,89E-01	6,87E+04	9,02E+07
200	1288,6	00:02	5,13E+00	1,40E-01	1,89E+04	4,14E+06

Tabuľka 28: Úlohy s exponenciálnou účelovou funkciou,
s počtom premenných $n = 55$, s počtom ohraničení $m = 60$
a s počtom iterácií metódy sečnicových nadrovín od 100 do 200

Komentár k Tabuľám 25, 26, 27, 28. S postupným zväčšovaním počtu iterácií metódy sečnicových nadrovín sa zväčšil počet simplexových iterácií, vzrástol čas výpočtu a zlepšili sa hodnoty presností. Dobrú presnosť riešenia sme dosiahli pre kvadratickú a bikvadratickú funkciu pri počte iterácii $MSN = 200$, pre logaritmickú funkciu pri $MSN = 2$. Vidíme, že

pre logaritmickú funkciu sme získali výbornú presnosť pri asi 80 simplexových iteráciách, zatiaľ čo pre kvadratickú a bikvadratickú bolo urobených približne 14000 simplexových iterácií a dosiahnutá presnosť bola menšia ako pre logaritmickú. Pre exponeciálnu funkciu sme pri $MSN = 200$ dosiahli presnosť $\varepsilon = 5,13$ a teda pre niektoré vygenerované úlohy bol zadaný počet iterácií $MSN = 200$ malý a bolo potrebných viac iterácií na dosiahnutie lepšej presnosti.

4.8 Zhrnutie experimentov

V prvom experimente sme sledovali vplyv rozmeru úloh na počet iterácií. Zistili sme, že so zväčšovaním počtu premenných a počtu ohraničení sa zväčšuje počet iterácií metódy sečnicových nadrovín, počet simplexových iterácií a čas výpočtu. Pre kvadratickú a bikvadratickú účelovú funkciu boli všetky výsledky podobné. Pre logaritmickú bolo urobených najmenej iterácii MSN (vo väčšine prípadoch len 2 iterácie) a čas výpočtu bol najmenší ("nulový"). Pre exponenciálnu funkciu neboli všetky úlohy vyriešené, čo asi spôsobili veľké čísla (10^{38}) a počítač nedokázal vyrátať úlohu. V druhom experimente sme sledovali vplyv počtu ohraničení na počet iterácií metódy sečnicových nadrovín. Zistili sme, že čím má úloha viac ohraničení vzhľadom na rovnaký počet premenných, tak na dosiahnutie žiadanej presnosti je vykonaných menej iterácií metódy sečnicových nadrovín. V treťom experimente sme sledovali vplyv požadovanej absolútnej presnosti na počet iterácií. Zistili sme, že pri absolútnej presnosti $\varepsilon < 0,001$ boli dosiahnuté veľmi dobré presnosti riešenia x^k a funkčnej hodnoty $f(x^k)$, zatiaľ čo dolný odhad z^k bol nepresný. V prvom experimente sme získali veľmi dobrý dolný odhad a teda usudzujeme, že "posledné" iterácie metódy sečnicových nadrovín poslúžili na vylepšenie dolného odhadu z^k. Vo štvrtom experimente sme sledovali akú presnosť riešenia získame pri vopred zadanom počte iterácií metódy sečnicových nadrovín. Týmto spôsobom sme získali veľmi dobrú presnosť pre logaritmickú a bikvadratickú funkciu, pre kvadratickú a exponenciálnu by bolo vhodnejšie použiť iné stopové kritérium.

Záver

V predloženej práci sme podrobne opísali metódu sečnicových nadrovín. Ukázali sme akým spôsobom rieši táto metóda úlohu konvexného programovania s lineárnymi ohraničeniami. Keďže v každej iterácii metódy sečnicových nadrovín riešime špeciálnu úlohu lineárneho programovania, venovali sme prvé dve kapitoly niektorým špecifickým otázkam riešenia úloh lineárneho programovania. Metóda sečnicových nadrovín je iteratívnou metódou a v práci sme ukázali postup v jednotlivých iteráciách a zhrnuli sme ho do algoritmu.

Na konkrétnom príklade v časti 3.4 sme krok za krokom predviedli postup pre riešenie úlohy s logaritmickou účelovou funkciou. V jazyku c++ sme programovo realizovali metódu sečnicových nadrovín pre úlohy konvexného programovania s lineárnymi ohraničeniami. Následne v numerických experimentoch sme preskúmali riešenie úloh s počtom premenných od 2 do 100 a s počtom ohraničení od 6 do 150 pre kvadratickú, bikvadratickú, logaritmickú a exponenciálnu účelovú funkciu. Pre všetky účelové funkcie bola metóda presná a aj pre úlohy s najväčším rozmerom sme dosiahli výbornú presnosť a dobrý čas výpočtu.

Literatúra

[1] Hamala, M. (1976): *Nelineárne programovanie*, Alfa, Bratislava.

[2] Plesník, J., Dupačová, J., Vlach, M. (1990): *Lineárne programovanie*, Alfa, Bratislava.

[3] Gass, Saul I. (1972): *Lineárne programovanie - Metódy a aplikácie*, Alfa, Bratislava.

[4] Kelley, J. E. (1960): *The Cutting-plane method for solving convex programs*, J. SIAM, Vol. 8, 703-712.

Príloha

Zdrojový kód programu Secnica

```
//------------------------------
#include <vcl.h>
#pragma hdrstop
#include <fstream.h>
#include<math.h>
#include <stdlib.h>
#include <stdio.h>
#include
<systdate.h>
#include "Unit1.h"
//------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"

TForm1 *Form1;
TMemo *memo;
TMemo *memo2;
TMemo *memo3;
TMemo *memo4;
TDateTime *Tim;
AnsiString cas=0;
const r=300;  // n+ m + 2 = r
const s=10000;
// presnost pre simplexovu tabulku
float eps=0.000000000001;
// pocet premenych a pocet nerovnic
int n,m;
int ran=10,ran2=5;
int PivotStlpec=0,PivotRiadok=0;
long double Pivot;
bool Ohranicenost=true;
int pocetiteracii=1;
int pocetsimpliteracii=0;
int w=0;
long double minfun=0;
//------------------------------
__fastcall TForm1::TForm1(TComponent*
  Owner): TForm(Owner) { }
//------------------------------
class Matica {
  public:
    long double a[r][r];
    void Vytvor();
    void VytvorA();
    void Vytvorkladn();
    void Vytvordiag();
};

class Vektor{
  public:
    long double a[r];
    // gradient fx
    Vektor Gradient(int n,Vektor x);
    void Vytvor(int roz);
};

// simplexova tabulka
class Tab{
  public:
    // ucelova funkcia
    long double n1[s];
    // tabulka[stlpec][riadok]
    long double a[s][r];
    int indexriadok[r];
    int indexstlpec[s];
};

void Vektor::Vytvor(int roz){
  for(int i=1;i<=roz;i++){
  a[i]=0;
  a[i]=random(ran);
  }
```

```
}

// generuje maticu
void Matica::Vytvor(){
  for(int j=1;j<=m;j++){
    for(int i=1;i<=n;i++){
     a[i][j]=0;
     a[i][j]=random(ran)-ran2;
    }
  }
}

// generuje maticu s kladnymi
   prvkami
void Matica::VytvorA(){
  for(int j=1;j<=m;j++){
    for(int i=1;i<=n;i++){
     a[i][j]=0;
     a[i][j]=random(ran)+1;
    }
  }
}

// generuje kladne definitnu
   symetricku maticu
void Matica::Vytvorkladn(){
  long double b[r][r];
  for(int i=1;i<=n;i++)
    for(int j=1;j<=n;j++)
      b[i][j]=random(ran)-ran2;
  for(int i=1;i<=n;i++){
    for(int j=1;j<=n;j++){
    a[i][j]=0;
    for(int k=1;k<=n;k++)
    a[i][j]=a[i][j]+b[i][k]*b[j][k];
    }
    a[i][i]++;
  }
}

// generuje diagonalnu maticu s
   kladnymi prvkami
void Matica::Vytvordiag(){
  for(int j=1;j<=n;j++){
    for(int i=1;i<=n;i++){
     a[i][j]=0;
     if(i==j)a[i][j]=random(ran)+1;
    }
  }
}

// logaritmus cez taylorov rozvoj
long double ln(int y){
  long double logaritmus=0;
  logaritmus=y-1-(1/2*pow(y-1,2))+
  (1/6*pow(y-1,3))-(1/24*pow(y-1,4))+
  (1/120*pow(y-1,5))-(1/720*pow(y-1,6));
  return logaritmus;
}
//------------------------------
// globalne premenne
Matica C;
Matica A;
Matica D;
Vektor p;
Vektor b;
Vektor xo;
Vektor x1;
Vektor Xopt;
// pre exponencialnu funkciu
Vektor c,d;
Tab T;
//------------------------------
// vypocet funkcnej hodnoty
long double
  FunkcnaHodnota(int n,Vektor x){
```

```
  // kvadraticka funkcia xCx + px
  long double fx=0,xCx=0,px=0;
  Vektor pom;
  for(int i=1;i<=n;i++){
    pom.a[i]=0;
    // vektor x * matica C
    for(int j=1;j<=n;j++)
      pom.a[i]=C.a[j][i]*x.a[j]+
               pom.a[i];
  }
  for(int i=1;i<=n;i++)
    xCx=pom.a[i]*x.a[i]+xCx;
  // vektor p * vektor x
  for(int i=1;i<=n;i++)
    px=p.a[i]*x.a[i]+px;
  // funkcna hodnota = xCx + px
  fx=xCx + px;
  return fx;
}

/*long double
    FunkcnaHodnota(int n,Vektor x){
  // bikvadraticka funkcia
  // (xDx)^2 + xCx + px
  long double fx=0,xCx=0,xDx=0,
              px=0,pom3=0;
  Vektor pom,pom2;
  long double omega=0;
  // omega
  for(int i=1;i<=n;i++){
    for(int j=1;j<=n;j++)
    omega=D.a[i][j]*D.a[i][j]+omega;
  }
  for(int i=1;i<=n;i++){
    pom.a[i]=0;
    pom2.a[i]=0;
    for(int j=1;j<=n;j++){
    // vektor x * matica C
    pom.a[i]=C.a[j][i]*x.a[j]+
             pom.a[i];
    // vektor x * matica D
    pom2.a[i]=D.a[j][i]*x.a[j]+
              pom2.a[i];
    }
  }
  for(int i=1;i<=n;i++){
    xCx=pom.a[i]*x.a[i]+xCx;
    xDx=pom2.a[i]*x.a[i]+xDx;
  }
  // (4*omega)^(-1) * (xDx)^2
  pom3=(xDx*xDx)/(4*omega);
  // vektor p * vektor x
  for(int i=1;i<=n;i++)
    px=p.a[i]*x.a[i]+px;
  fx=pom3 + xCx + px;
  return fx;
}   */

/*long double
  FunkcnaHodnota(int n,Vektor x){
  // logaritmicka funkcia sum(x lnx)+px
  long double fx=0,px=0;
  long double xlnx=0,pom=0;
  // logaritmus cez taylorov rozvoj
  // sum (x * ln x )
  for(int i=1;i<=n;i++){
    if(x.a[i]>0.2)
      pom=x.a[i]*log(x.a[i])+pom;
    else
      pom=x.a[i]*ln(x.a[i])+pom;
  }
  xlnx=pom;
  // vektor p * vektor x
  for(int i=1;i<=n;i++)
    px=p.a[i]*x.a[i]+px;
  // funkcna hodnota = xlnx + px
```

```
  fx=xlnx + px;
  return fx;
} */

/*long double
  FunkcnaHodnota(int n,Vektor x){
  // exponencialna funkcia
  // sum(exp(cx + d)) + px
  long double fx=0,px=0,expx=0,pom=0;
  // sum(exp(c*x + d))
  for(int i=1;i<=n;i++){
    pom=exp(c.a[i]*x.a[i]+d.a[i])+pom;
  }
  expx=pom;
  // vektor p * vektor x
  for(int i=1;i<=n;i++)
    px=p.a[i]*x.a[i]+px;
  // funkcna hodnota = expx + px
  fx=expx + px;
  return fx;
}    */
//------------------------------
// vypocet gradientu funkcie
Vektor Vektor::
Gradient(int n,Vektor x){
  // kvadraticka funkcia
  Vektor pom;
  Vektor g;
  for(int i=1;i<=r;i++){
    pom.a[i]=0;
    g.a[i]=0;
  }
  for(int i=1;i<=n;i++){
  pom.a[i]=0;
  for(int j=1;j<=n;j++)
  pom.a[i]=C.a[j][i]*x.a[j]+pom.a[i];
  }
  // gradient=
     2*vektor x* matica C +vektor p
  for(int i=1;i<=n;i++)
    g.a[i]=2*pom.a[i]+p.a[i];
  return g;
}

/*Vektor Vektor::Gradient(int n,Vektor x){
  // bikvadraticka funkcia
  Vektor pom,pom2;
  Vektor g;
  long double xDx=0;
  long double omega=0;
  for(int i=1;i<=n;i++){
    for(int j=1;j<=n;j++)
    omega=D.a[i][j]*D.a[i][j]+omega;
  }
  // vektor x * matica C
  for(int i=1;i<=n;i++){
  pom.a[i]=0;
  for(int j=1;j<=n;j++)
  pom.a[i]=C.a[j][i]*x.a[j]+pom.a[i];
  }
  // vektor x * matica D
  for(int i=1;i<=n;i++){
    pom2.a[i]=0;
    for(int j=1;j<=n;j++)
    pom2.a[i]=D.a[j][i]*x.a[j]+
              pom2.a[i];
  }
  // x*D*x
  for(int i=1;i<=n;i++)
    xDx=pom2.a[i]*x.a[i]+xDx;
// g =(D*x*x*D*x)/(omega)+2*C*x+ p
  for(int i=1;i<=n;i++)
    g.a[i]=(pom2.a[i]*xDx)/(omega)+
           2*pom.a[i]+p.a[i];
  return g;
}    */
```

```
/*Vektor Vektor::
Gradient(int n,Vektor x){
  // logaritmicka funkcia
  Vektor pom;
  Vektor g;
  for(int i=1;i<=n;i++){
    pom.a[i]=0;
    g.a[i]=0;
  }
  // logaritmus cez taylorov rozvoj
  // gradient = ln x + 1
  for(int i=1;i<=n;i++){
    if(x.a[i]>0.2)
      pom.a[i]=log(x.a[i])+1;
    else
      pom.a[i]=ln(x.a[i])+1;
  }
// gradient=vektor ln x+1+vektor p
  for(int i=1;i<=n;i++)
    g.a[i]=pom.a[i]+p.a[i];
  return g;
}  */

/*Vektor Vektor::
Gradient(int n,Vektor x){
  // exponencialna funkcia
  Vektor pom;
  Vektor g;
  for(int i=1;i<=n;i++){
    pom.a[i]=0;
    g.a[i]=0;
  }
  // gradient = exp(cx+d)*c
  for(int i=1;i<=n;i++){
    pom.a[i]=exp(c.a[i]*x.a[i]+
             d.a[i])*c.a[i];
  }
  // gradient=vektor exp(cx+d)*c+
             vektor p
  for(int i=1;i<=n;i++)
    g.a[i]=pom.a[i]+p.a[i];
  return g;
}  */
//------------------------------
long double Velkost(Vektor x){
  long double pom=0;
  for(int i=1;i<=n;i++)
    pom=x.a[i]*x.a[i]+pom;
  pom=sqrt(pom);
  return pom;
}

long double Gama(Vektor x){
  long double gama=0,gx=0;
  Vektor grad;
  for(int i=1;i<=n;i++)
    gx=(grad.Gradient(n,x).a[i])*
       x.a[i]+gx;
  gama=gx-FunkcnaHodnota(n,x);
  return gama;
}

long double Vzdialenost(Vektor x){
  long double pom=0;
  for(int i=1;i<=n;i++)
    pom=(Xopt.a[i]-x.a[i])*
        (Xopt.a[i]-x.a[i])+pom;
  return sqrt(pom);
}

long double Delta(int n,Vektor x){
  long double delta=0;
  delta=FunkcnaHodnota(n,x)+T.n1[0];
  return delta;
}
```

```
//------------------------------
// zostavi simplexovu tabulku
void ZostavTabulku(){
  // nulovanie tabulky
  for(int i=0;i<s;i++){
    T.n1[i]=0;
    for(int j=0;j<r;j++)
      T.a[i][j]=0;
  }
  // prava strana
  for(int j=1;j<=n;j++)
    T.a[0][j]=0;
  T.a[0][n+1]=1;
  // matica I
  for(int j=1;j<=n+1;j++){
    for(int i=1;i<=n+1;i++){
    if(i==j)
      T.a[i][j]=1;
    else T.a[i][j]=0;
    }
  }
  // matica -A transponovana
  for(int j=1;j<=n;j++){
    for(int i=1;i<=m;i++){
    T.a[i+n+1][j]=-A.a[j][i];
    T.a[i+n+1][n+1]=0;
    }
  }
  Vektor grad2;
  Vektor grad1;
  for(int i=0;i<r;i++){
    grad1.a[i]=0;
    grad2.a[i]=0;
  }
  // gradient=2*x*C + p
  grad2=grad1.Gradient(n,xo);
// gradient v poslednom stlpci tabulky
  for(int j=1;j<=n;j++){
    T.a[n+m+2][j]=-grad2.a[j];
  T.a[n+m+2][n+1]=1;
  }
}

// najdi pivota pre ulohu na min,
// odstraneny stlpec omega
void NajdiPivota2(){
  bool JePivot=false;
  int PocetZapornych=0;
  long double min=0;
  long double minocen=0;
  for(int i=1;i<=n+m+2+w;i++){
    PocetZapornych=0;
    if(T.n1[i]>eps)
      for(int k=1;k<=n+1;k++)
        if(T.a[i][k]<=-eps)
          PocetZapornych++;
    if(PocetZapornych==n+1)
      Ohranicenost=false;
  }
  if(Ohranicenost==true){
    for(int j=1;j<=n+m+2+w;j++){
      if(T.n1[j]>minocen){
        if(j!=n+1){
          minocen=T.n1[j];
          PivotStlpec=j;
        }
      }
    }
    for(int i=1;i<=n+1;i++){
      if(T.a[PivotStlpec][i]>eps&&
      JePivot==true &&
      T.a[0][i]/T.a[PivotStlpec][i]<min)
      {
      min=T.a[0][i]/T.a[PivotStlpec][i];
      PivotRiadok=i;
      Pivot=T.a[PivotStlpec][i];
```

```
      }
      if(T.a[PivotStlpec][i]>eps&&
      JePivot==false)
      {
      min=T.a[0][i]/T.a[PivotStlpec][i];
      PivotRiadok=i;
      Pivot=T.a[PivotStlpec][i];
      JePivot=true;
      }
    }
  }
  if(Ohranicenost==false){
    for(int i=1;i<=n+m+2+w;i++){
      if(T.n1[i]>eps)
      if(i!=n+1){
        for(int k=1;k<=n+1;k++)
          if(T.a[i][k]>eps){
            PivotStlpec=i;
            PivotRiadok=k;
            Pivot=T.a[i][k];
            JePivot=true;
            Ohranicenost=true;
          }
      }
    }
  }
}

// najdi pivota pre ulohu na min
void NajdiPivota3(){
  bool JePivot=false;
  bool PravaSNula=false;
  int PocetZapornych=0;
  long double min=0;
  long double minocen=0;
  long double maxcislo=0;
  for(int i=1;i<=n+m+2+w;i++){
    PocetZapornych=0;
    if(T.n1[i]>eps)
      for(int k=1;k<=n+1;k++)
        if(T.a[i][k]<=-eps)
          PocetZapornych++;
    if(PocetZapornych==n+1)
      Ohranicenost=false;
  }
  if(Ohranicenost==true){
    for(int j=1;j<=n+m+2+w;j++){
      if(T.n1[j]>minocen){
        minocen=T.n1[j];
        PivotStlpec=j;
      }
    }
    for(int i=1;i<=n+1;i++){
      if(T.a[0][i]==0 &&
      T.a[PivotStlpec][i]>eps)
        PravaSNula=true;
    }
    if(PravaSNula==true){
      for(int i=1;i<=n+1;i++){
        if(T.a[PivotStlpec][i]>eps&&
        T.a[0][i]==0&&
        T.a[PivotStlpec][i]>maxcislo)
        {
        maxcislo=T.a[PivotStlpec][i];
        PivotRiadok=i;
        Pivot=T.a[PivotStlpec][i];
        }
      }
    }
    else{
      for(int i=1;i<=n+1;i++){
      if(T.a[PivotStlpec][i]>eps&&
      JePivot==true&&
      T.a[0][i]/T.a[PivotStlpec][i]<min)
      {
      min=T.a[0][i]/T.a[PivotStlpec][i];
```

```
      PivotRiadok=i;
      Pivot=T.a[PivotStlpec][i];
      }
      if(T.a[PivotStlpec][i]>eps&&
      JePivot==false)
      {
      min=T.a[0][i]/T.a[PivotStlpec][i];
      PivotRiadok=i;
      Pivot=T.a[PivotStlpec][i];
      JePivot=true;
      }
      }
    }
  }
  if(Ohranicenost==false){
    for(int i=1;i<=n+m+2+w;i++){
      if(T.n1[i]>eps)
        for(int k=1;k<=n+1;k++)
          if(T.a[i][k]>eps){
            PivotStlpec=i;
            PivotRiadok=k;
            Pivot=T.a[i][k];
            JePivot=true;
            Ohranicenost=true;
          }
    }
  }
}

// urobi jednu transformaciu v
// simplexovej tabulke
void Krok(){
  long double pom=0;
  for(int i=0;i<=n+m+2+w;i++)
    T.a[i][PivotRiadok]=
    T.a[i][PivotRiadok]/Pivot;
  for(int j=1;j<=n+1;j++){
    pom=-T.a[PivotStlpec][j];
      if(j!=PivotRiadok)
        for(int i=0;i<=n+m+2+w;i++){
          long double pom1,pom2;
          pom1=T.a[i][PivotRiadok]*pom;
          pom2=T.a[i][j];
          pom1+=pom2;
          T.a[i][j]=pom1;
        }
    }
    pom=-T.n1[PivotStlpec];
    for(int i=0;i<=n+m+2+w;i++){
      long double pom1,pom2;
      pom1=T.a[i][PivotRiadok]*pom;
      pom2=T.n1[i];
      pom1+=pom2;
      T.n1[i]=pom1;
    }
  }

// zmeni indexy v riadkoch pri
// transformacii simplexovej tabulky
void Zamen(int pivotriad,int pivotstlp){
  T.indexriadok[pivotriad]=
  T.indexstlpec[pivotstlp];
}

// hlada kladne prvky
bool ExistujeKladne(){
  bool pom=true;
  for(int i=1;i<=n+m+2+w;i++)
    if(i!=n+1){
      pom=(pom && T.n1[i]<=eps);
    }
  return !pom;
}
//------------------------------
// transformacia simplexovej
// tabulky s pomocnou premennou
```

```
void PrvaFaza(){
// pomocna premenna omega
  T.n1[n+1]=-1;
// eliminuje pomocnu premennu
  for(int i=0;i<=n+m+2;i++){
    long double pom1=0;
    pom1=T.a[i][n+1]+T.n1[i];
    T.n1[i]=pom1;
  }
}

// transformacia simplexovej tab.
// po dodani ucelovej funkcie
void DruhaFaza(){
  Vektor cB;
  Vektor cN;
  Vektor pom;
// povodna ucelova funkcia DU
  Vektor bgama;
  for(int j=0;j<=r;j++){
    cB.a[j]=0;
    cN.a[j]=0;
    pom.a[j]=0;
  }
  for(int j=0;j<=r;j++)
    bgama.a[j]=b.a[j];
// gama v zaciatocnom bode xo
  bgama.a[m+1]=Gama(xo);
// vyberie zo simplexovej tabulky
// cB z ucelovej funkcie b+gama
for(int j=1;j<=n+1;j++){
if(T.indexriadok[j]>n+1)
cB.a[j]=bgama.a[T.indexriadok[j]-n-1];
  }
// vyberie cN
  for(int i=1;i<=m+1;i++){
    cN.a[i+n+1]=bgama.a[i];
  }
  for(int i=1;i<=n+m+2;i++){
    for(int j=1;j<=n+1;j++){
    if(T.indexstlpec[i]==
       T.indexriadok[j])
    cN.a[i]=0;
    }
  }
// vynuluje riadok ucelovej funkcie
  for(int i=0;i<=n+m+2;i++)
    T.n1[i]=0;
  for(int i=0;i<=n+1+m+1;i++){
    pom.a[i]=0;
    for(int j=1;j<=n+1;j++){
// vynasobi stlpce s vektorom cB
    pom.a[i]=T.a[i][j]*cB.a[j]+
             pom.a[i];
    }
// doda do ucelovej funkcie
// cb * stlpce tabulky
    T.n1[i]=pom.a[i];
  }
  for(int i=1;i<=n+m+2;i++){
  for(int j=1;j<=n+1;j++){
// najde bazicke premenne
  if(T.indexstlpec[i]==T.indexriadok[j])
// do stlpcov bazickych premennych
// pridu do ucelovej funkcie nuly
  T.n1[i]=0;
  }
  }
  for(int i=1;i<=n+m+2;i++){
// vyratanie riadka ucelovej funkcie
  T.n1[i]=T.n1[i]-cN.a[i];
  }
}

// ratanie stlpca v1 a hodnoty gama
// a pridanie do simplexovej tabulky
```

```
void TretiaFaza(){
  Vektor grad4;
  Vektor pom2;
  Vektor v1;
  long double pom21=0;
  for(int j=0;j<=r;j++){
    grad4.a[j]=0;
    pom2.a[j]=0;
    v1.a[j]=0;
  }
// pridanie dalsieho ohranicenia
  w=w+1;
  pocetiteracii++;
// gradient=2*x1*C+P
  for(int i=1;i<=n;i++)
  grad4.a[i]=
    -(grad4.Gradient(n,x1)).a[i];
  grad4.a[n+1]=1;
// prenasobenie gradienta rozsirenou
// inverznou bazou
  for(int j=1;j<=n+1;j++){
    pom2.a[j]=0;
    for(int i=1;i<=n+1;i++)
    pom2.a[j]=T.a[i][j]*grad4.a[i]+
              pom2.a[j];
  }
  for(int i=1;i<=n+1;i++){
    pom21=T.n1[i]*grad4.a[i]+pom21;
  }
// gama1+w*g = hodnota v poslednom
// riadku stlpca v1
  T.n1[n+m+2+w]=-Gama(x1)+pom21;
// stlpec premenej v1
  v1=pom2;
  for(int j=1;j<=n+1;j++)
    T.a[n+m+2+w][j]=v1.a[j];
}
//------------------------------
```

```
void ZobrazTabulku(){
  memo=Form1->Memo1;
  memo2=Form1->Memo2;
  Vektor gradient;
  for(int i=1;i<=r;i++)
    gradient.a[i]=0;
  for(int i=1;i<=n;i++)
    gradient.a[i]=
    (gradient.Gradient(n,x1)).a[i];
  AnsiString medzera;
  for(int j=1;j<=n+1;j++){
    medzera=" ";
    for(int i=0;i<=n+m+2+w;i++){
      medzera=medzera +
      FloatToStr(T.a[i][j])+"\t";
    }
    memo->Lines->Add(medzera);
  }
  memo->Lines->Add(" ");
  medzera=" ";
  for(int i=0;i<=n+m+2+w;i++){
    medzera=medzera+
    FloatToStr(T.n1[i])+"\t";
  }
  memo->Lines->Add(medzera);
  memo->Lines->Add(" ");
  AnsiString medzera2;
medzera2=" ";
medzera2=medzera2 +
  FloatToStr(pocetiteracii)+"\t";
medzera2=medzera2 +
FloatToStr(pocetsimpliteracii)+"\t";
medzera2=medzera2 +
  cas+"\t";
medzera2=medzera2 +
  FloatToStr(Vzdialenost(x1))+"\t";
medzera2=medzera2 +
  FloatToStr(Vzdialenost(x1)/
```

```
    (1+Velkost(Xopt)))+"\t";
medzera2=medzera2 +
  FloatToStr(-T.n1[0])+"\t";
medzera2=medzera2 +
FloatToStr(FunkcnaHodnota(n,x1))
  +"\t";
medzera2=medzera2 +
FloatToStr(FunkcnaHodnota(n,Xopt))
  +"\t";
//medzera2=medzera2 +
//  FloatToStr(Velkost(gradient))
  +"\t";
//medzera2=medzera2 +
//  FloatToStr(gama(x1))+"\t";
//medzera2=medzera2 +
//  FloatToStr(grad_krat_x)+"\t";
memo2->Lines->Add(medzera2);
memo2->Lines->Add(" "); }
//------------------------------
void __fastcall TForm1::
FormCreate(TObject *Sender) {
  randomize();
  for(int i=1;i<=r;i++){
    xo.a[i]=0;
    x1.a[i]=0;
    Xopt.a[i]=0;
    p.a[i]=0;
    b.a[i]=0;
    c.a[i]=0;
    d.a[i]=0;
    for(int j=1;j<=r;j++){
      C.a[i][j]=0;
      A.a[i][j]=0;
      D.a[i][j]=0;
    }
  }
}
//------------------------------
void __fastcall TForm1::
Button1Click(TObject *Sender) {
// generator uloh
  memo3=Form1->Memo3;
  n=25;m=30;
// matica C symetricka kladne def.
  C.Vytvorkladn();
// matica A kladna
  A.VytvorA();
// optimalne riesenie x kladne
  Xopt.Vytvor(n);
////////////////////////////////
//  pre kvadraticku
  Vektor g;
  Vektor pom;
  for(int i=1;i<=n;i++){
    pom.a[i]=0;
    for(int j=1;j<=n;j++)
    pom.a[i]=C.a[j][i]*Xopt.a[j]+
             pom.a[i];
  }
// gradient=2*vektor Xopt*matica C
  for(int i=1;i<=n;i++)
    g.a[i]=2*pom.a[i];
////////////////////////////////
//  pre bikvadraticku
/*  D.Vytvordiag();
  long double xDx=0,omega=0;
  Vektor Cx,Dx;
  for(int i=1;i<=n;i++){
    for(int j=1;j<=n;j++)
    omega=D.a[i][j]*D.a[i][j]+omega;
  }
  Vektor g;
  for(int i=1;i<=n;i++){
    Cx.a[i]=0;
    Dx.a[i]=0;
    for(int j=1;j<=n;j++){
```

```
// vektor Xopt *matica C
    Cx.a[i]=C.a[j][i]*Xopt.a[j]+
            Cx.a[i];
// vektor Xopt *matica D
    Dx.a[i]=D.a[j][i]*Xopt.a[j]+
            Dx.a[i];
    }
  }
// vektor Xopt *vektor Dx
  for(int i=1;i<=n;i++)
    xDx=Dx.a[i]*Xopt.a[i]+xDx;
// gradient=(xDx/omega)*vektor Dx+
//          2*vektor Xopt*matica C
  for(int i=1;i<=n;i++)
    g.a[i]=(xDx/omega)*Dx.a[i]+
           2*Cx.a[i];*/
////////////////////////////////
//  pre logaritmicku
/*  Vektor pom;
  Vektor g;
  for(int i=1;i<=r;i++){
    pom.a[i]=0;
    g.a[i]=0;
  }
// logaritmus cez taylorov rozvoj
  for(int i=1;i<=n;i++){
    if(Xopt.a[i]>0.2)
      pom.a[i]=log(Xopt.a[i])+1;
    else
      pom.a[i]=ln(Xopt.a[i])+1;
  }
// gradient=vektor ln x + 1
  for(int i=1;i<=n;i++)
    g.a[i]=pom.a[i];  */
////////////////////////////////
//  pre exponencialnu
/*  Vektor pom;
  Vektor g;
  c.Vytvor(n);
  d.Vytvor(n);
  for(int i=1;i<=n;i++){
    c.a[i]=c.a[i]/10;
    d.a[i]=d.a[i]/10;
  }
  for(int i=1;i<=r;i++){
    pom.a[i]=0;
    g.a[i]=0;
  }
  for(int i=1;i<=n;i++){
    pom.a[i]=exp(c.a[i]*Xopt.a[i]+
             d.a[i])*c.a[i];
  }
// gradient=vektor exp(cx+d)*c
  for(int i=1;i<=n;i++)
    g.a[i]=pom.a[i];  */
////////////////////////////////
  Vektor u,v,y;
  u.Vytvor(m);
  v.Vytvor(n);
  y.Vytvor(m);
  u.a[0]=0;
  v.a[0]=0;
  y.a[0]=0;
  for(int i=m+1;i<=r;i++){
    u.a[i]=0;
    y.a[i]=0;
  }
  for(int i=n+1;i<=r;i++)
    v.a[i]=0;
  for(int i=1;i<=n;i++){
    if(Xopt.a[i]>0)
      v.a[i]=0;
    if(Xopt.a[i]==0)
      v.a[i]=1;
  }
  for(int i=1;i<=m;i++){
```

```
    if(u.a[i]>0)
      y.a[i]=0;
    if(u.a[i]==0)
      y.a[i]=1;
  }
  Vektor pom2;
  for(int i=1;i<=n;i++){
  pom2.a[i]=0;
  for(int j=1;j<=m;j++)
    pom2.a[i]=A.a[i][j]*u.a[j]+
              pom2.a[i];
  }
// vyratanie vektora p
  for(int i=1;i<=n;i++)
    p.a[i]=v.a[i]-g.a[i]-pom2.a[i];
  Vektor pom3;
  for(int i=1;i<=m;i++){
  pom3.a[i]=0;
  for(int j=1;j<=n;j++)
  pom3.a[i]=A.a[j][i]*Xopt.a[j]+
            pom3.a[i];
  }
// vyratanie vektora b
  for(int j=1;j<=m;j++)
    b.a[j]=pom3.a[j]+y.a[j];
// vypis velicin do mema
  AnsiString medzera3;
  medzera3=" ";
  for(int j=1;j<=m;j++){
    for(int i=1;i<=n;i++){
      medzera3=medzera3 +
        FloatToStr(A.a[i][j])+"\t";
    }
  memo3->Lines->Add(medzera3);
  medzera3=" ";
  }
  memo3->Lines->Add(medzera3);
  for(int j=1;j<=n;j++){
    for(int i=1;i<=n;i++){
      medzera3=medzera3 +
        FloatToStr(C.a[i][j])+"\t";
    }
  memo3->Lines->Add(medzera3);
  medzera3=" ";
  }
  memo3->Lines->Add(medzera3);
  for(int j=1;j<=n;j++){
    medzera3=medzera3 +
      FloatToStr(Xopt.a[j])+"\t";
  memo3->Lines->Add(medzera3);
  medzera3=" ";
  }
  memo3->Lines->Add(medzera3);
  for(int j=1;j<=n;j++){
    medzera3=medzera3 +
    FloatToStr(v.a[j])+"\t";
  memo3->Lines->Add(medzera3);
  medzera3=" ";
  }
  memo3->Lines->Add(medzera3);
  for(int j=1;j<=m;j++){
    medzera3=medzera3 +
    FloatToStr(u.a[j])+"\t";
  memo3->Lines->Add(medzera3);
  medzera3=" ";
  }
  memo3->Lines->Add(medzera3);
  for(int j=1;j<=m;j++){
    medzera3=medzera3 +
    FloatToStr(y.a[j])+"\t";
  memo3->Lines->Add(medzera3);
  medzera3=" ";
  }
  memo3->Lines->Add(medzera3);
  for(int j=1;j<=n;j++){
    medzera3=medzera3 +
```

```
    FloatToStr(p.a[j])+"\t";
  memo3->Lines->Add(medzera3);
  medzera3=" ";
  }
  memo3->Lines->Add(medzera3);
  for(int j=1;j<=m;j++){
    medzera3=medzera3 +
      FloatToStr(b.a[j])+"\t";
  memo3->Lines->Add(medzera3);
  medzera3=" ";
  }
  memo3->Lines->Add(medzera3);
  medzera3=medzera3 +
    FloatToStr(m)+"\t";
  memo3->Lines->Add(medzera3);
  medzera3=" ";
  memo3->Lines->Add(medzera3);
  medzera3=medzera3 +
    FloatToStr(n)+"\t";
  memo3->Lines->Add(medzera3);
  medzera3=" ";
// matica D, omega
/*  memo3->Lines->Add(medzera3);
  for(int j=1;j<=n;j++){
    for(int i=1;i<=n;i++){
    medzera3=medzera3 +
      FloatToStr(D.a[i][j])+"\t";
    }
  memo3->Lines->Add(medzera3);
  medzera3=" ";
  }
  memo3->Lines->Add(medzera3);
  medzera3=medzera3 +
    FloatToStr(omega)+"\t";
  memo3->Lines->Add(medzera3);
  medzera3=" ";   */
}
//---------------------------
```

```
void __fastcall TForm1::
Button2Click(TObject *Sender){
// rata startovaci bod x0,
// Min{ grad_f(0)*x|Ax<=b,x>=0 }
  memo4=Form1->Memo4;
// prava strana
  for(int j=1;j<=m;j++)
    T.a[0][j]=b.a[j];
// matica I
  for(int j=1;j<=m;j++){
    for(int i=1;i<=m;i++){
      if(i==j)
        T.a[i][j]=1;
      else
        T.a[i][j]=0;
    }
  }
  // matica A
  for(int j=1;j<=m;j++){
    for(int i=1;i<=n;i++){
      T.a[i+m][j]=A.a[i][j];
    }
  }
  for(int j=1;j<=m;j++){
    T.indexriadok[j]=j;
  }
  for(int i=1;i<=m+n;i++){
    T.indexstlpec[i]=i;
  }
  Vektor g;
  Vektor grad6;
  Vektor nulovy;
  for(int i=0;i<=r;i++)
    nulovy.a[i]=0;
  g = grad6.Gradient(n,nulovy);
  for(int i=1;i<=m+n;i++){
    T.n1[i+m]=g.a[i];
  }
```

```
  while(ExistujeKladne()==true &
        Ohranicenost==true){
    NajdiPivota2();
    if(Ohranicenost==true)
    Krok();
    Zamen(PivotRiadok,PivotStlpec);
  }
  if(Ohranicenost==false)
    ShowMessage("Neohranicenost");
  for(int i=0;i<=n;i++)
    xo.a[i]=0;
  for(int j=1;j<=m;j++){
  if(T.indexriadok[j]>=m+1&&
     T.indexriadok[j]<=m+n)
// priradi do xo bazicke riesenie
  xo.a[T.indexriadok[j]-m]=T.a[0][j];
  }
  AnsiString medzera4;
  medzera4=" ";
  for(int i=1;i<=n;i++){
    medzera4=medzera4 +
      FloatToStr(xo.a[i]);
  memo4->Lines->Add(medzera4);
  medzera4=" ";
  }
}
//------------------------------
void __fastcall TForm1::
Button3Click(TObject *Sender) {
  TDateTime start=0, koniec=0;
  start = Tim->CurrentTime();
// rata po presnost
  long double epsilonpom, epsilon;
  epsilonpom=StrToInt(Edit2->Text);
// epsilon = 0,1 ^ epsilonpom
  epsilon = pow(0.1,epsilonpom);
// zostavi a zobrazi prvu
// simplexovu tabulku
  for(int j=1;j<=n+1;j++)
    T.indexriadok[j]=j;
  for(int i=1;i<=n+1+m+1+w;i++){
    T.indexstlpec[i]=i;
  }
  ZostavTabulku();
// transformacia simplexovej tabulky
// s pomocnou premennou
  PrvaFaza();
  while(ExistujeKladne()==true&&
        Ohranicenost==true){
    NajdiPivota3();
    if(Ohranicenost==true)
    Krok();
    pocetsimpliteracii++;
    Zamen(PivotRiadok,PivotStlpec);
    for(int i=0;i<=r;i++)
      x1.a[i]=0;
// vyberie zo simplexovej
// tabulky optimalne x
    for(int i=1;i<=n;i++)
      x1.a[i]=-T.n1[i];
  }
  if(Ohranicenost==false)
    ShowMessage("Neohranicenost");
// transformacia simplexovej tabulky
// s povodnou ucelovou funkciou
  DruhaFaza();
  for(int i=0;i<=r;i++)
    x1.a[i]=0;
  for(int i=1;i<=n;i++)
    x1.a[i]=-T.n1[i];
  while(ExistujeKladne()==true&&
        Ohranicenost==true){
    NajdiPivota2();
    if(Ohranicenost==true)
    Krok();
    pocetsimpliteracii++;
```

```
    Zamen(PivotRiadok,PivotStlpec);
      for(int i=0;i<=r;i++)
      x1.a[i]=0;
// vyberie zo simplexovej
// tabulky optimalne x
    for(int i=1;i<=n;i++)
    x1.a[i]=-T.n1[i];
  }
  if(Ohranicenost==false)
    ShowMessage("Neohranicenost");

// while(Vzdialenost(x1)>epsilon){
  while(Delta(n,x1) > epsilon){
  TretiaFaza();
    while(ExistujeKladne()==true&&
          Ohranicenost==true){
      NajdiPivota2();
      if(Ohranicenost==true)
      Krok();
      pocetsimpliteracii++;
      Zamen(PivotRiadok,PivotStlpec);
    }
    for(int i=0;i<=r;i++)
      x1.a[i]=0;
    for(int i=1;i<=n;i++)
      x1.a[i]=-T.n1[i];
  }
  koniec = Tim->CurrentTime();
  cas=koniec-start;
  ZobrazTabulku();
}
//------------------------------
void __fastcall TForm1::
Button4Click(TObject *Sender) {
//  urobi zadany pocet iteracii
  TDateTime start=0, koniec=0;
  start = Tim->CurrentTime();
// zostavi a zobrazi prvu
// simplexovu tabulku
  for(int j=1;j<=n+1;j++)
    T.indexriadok[j]=j;
  for(int i=1;i<=n+1+m+1+w;i++){
    T.indexstlpec[i]=i;
  }
  ZostavTabulku();
// transformacia simplexovej tabulky
// s pomocnou premennou
  PrvaFaza();
  while(ExistujeKladne()==true&&
        Ohranicenost==true){
    NajdiPivota3();
    if(Ohranicenost==true)
      Krok();
      pocetsimpliteracii++;
      Zamen(PivotRiadok,PivotStlpec);
// vyberie zo simplexovej
// tabulky optimalne x
      for(int i=0;i<=r;i++)
        x1.a[i]=0;
      for(int i=1;i<=n;i++)
        x1.a[i]=-T.n1[i];
  }
  if(Ohranicenost==false)
    ShowMessage("Neohranicenost");
// transformacia simplexovej tabulky
// s povodnou ucelovou funkciou
  DruhaFaza();
  for(int i=0;i<=r;i++)
    x1.a[i]=0;
  for(int i=1;i<=n;i++)
    x1.a[i]=-T.n1[i];
  while(ExistujeKladne()==true&&
        Ohranicenost==true){
    NajdiPivota2();
    if(Ohranicenost==true)
      Krok();
```

```
      pocetsimpliteracii++;
      Zamen(PivotRiadok,PivotStlpec);
      for(int i=0;i<=r;i++)
        x1.a[i]=0;
// vyberie zo simplexovej
// tabulky optimalne x
      for(int i=1;i<=n;i++)
        x1.a[i]=-T.n1[i];
  }
  if(Ohranicenost==false)
    ShowMessage("Neohranicenost");

// urobi zadany pocet iteracii
  int i = StrToInt(Edit1->Text);
  int j=2;
  while(j<=i){
    j++;
    TretiaFaza();
    while(ExistujeKladne()==true&&
          Ohranicenost==true){
      NajdiPivota2();
      if(Ohranicenost==true)
      Krok();
      pocetsimpliteracii++;
      Zamen(PivotRiadok,PivotStlpec);
    }
    for(int i=0;i<=r;i++)
      x1.a[i]=0;
    for(int i=1;i<=n;i++)
      x1.a[i]=-T.n1[i];
  }
  koniec=Tim->CurrentTime();
  cas=koniec-start;
  ZobrazTabulku();
  if(Ohranicenost==false)
    ShowMessage("Neohranicenost");
}
//------------------------------
```